IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

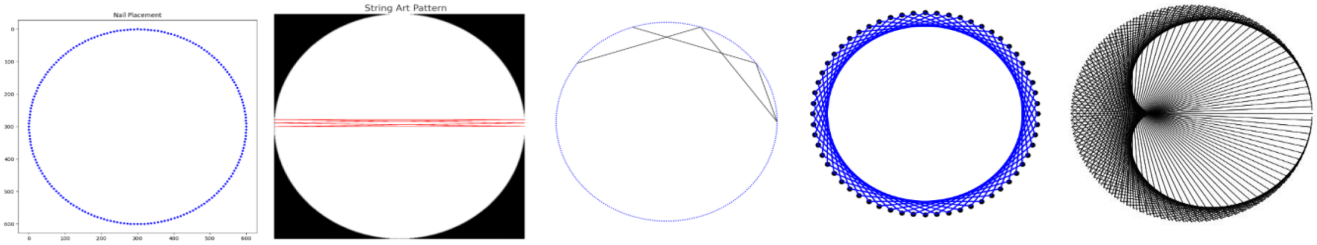


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

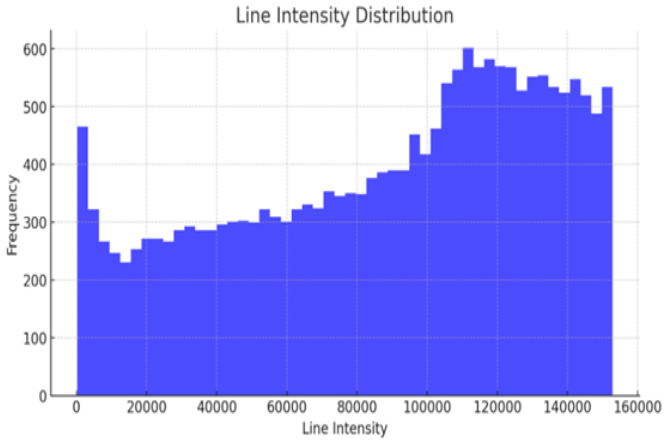


FIGURE 2: Line intensity distribution.

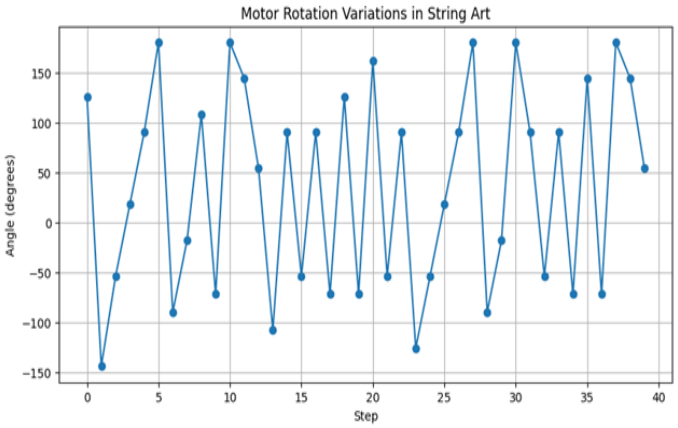


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

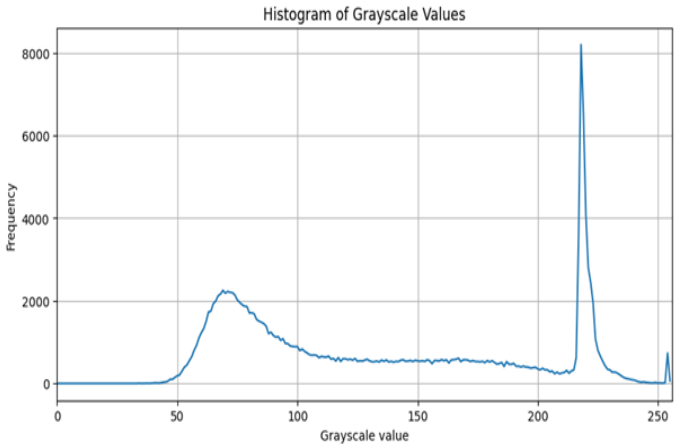


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

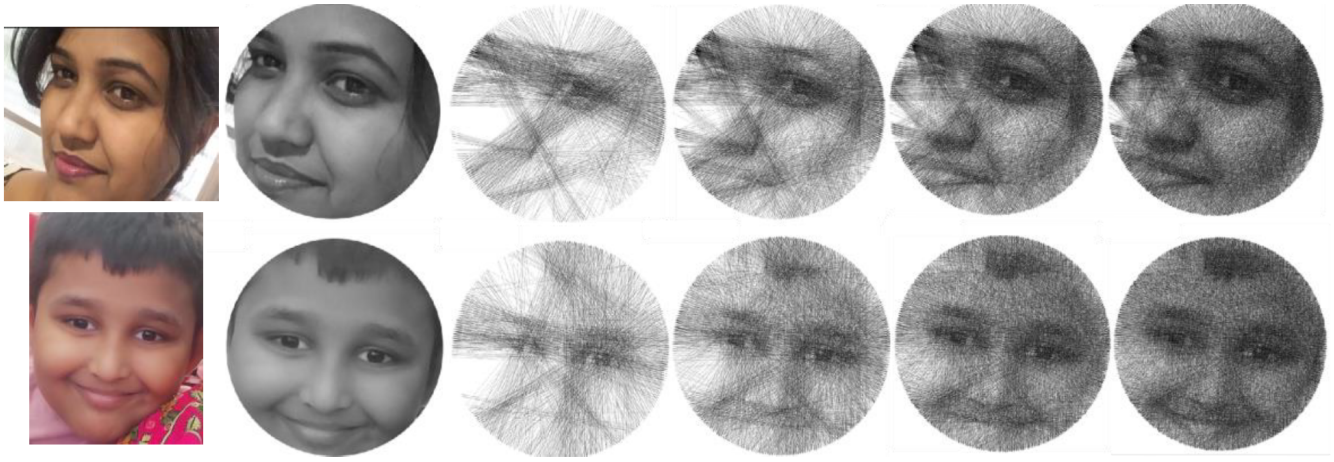


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

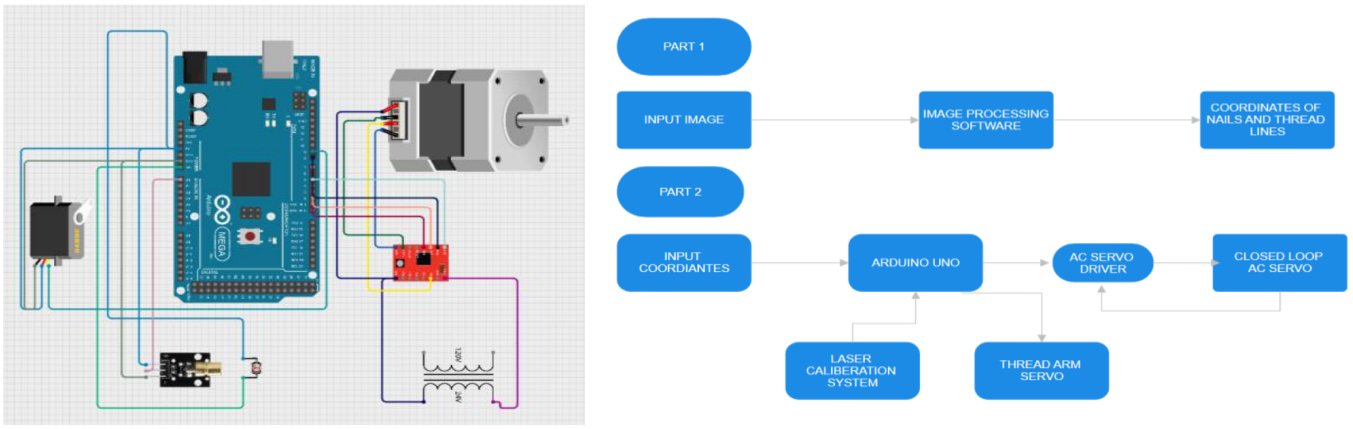


FIGURE 7: Circuit connection with block diagram representation.

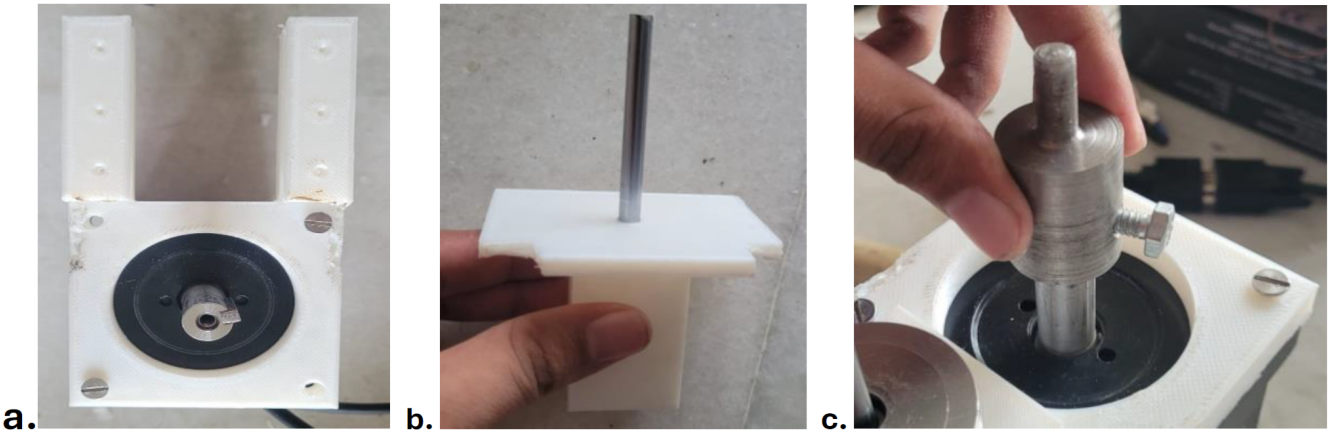


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

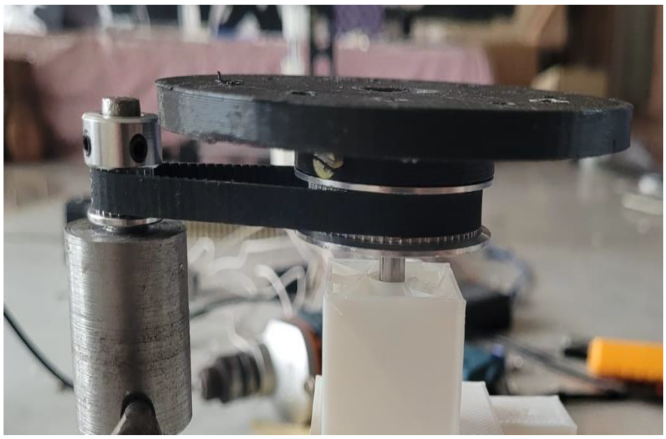


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

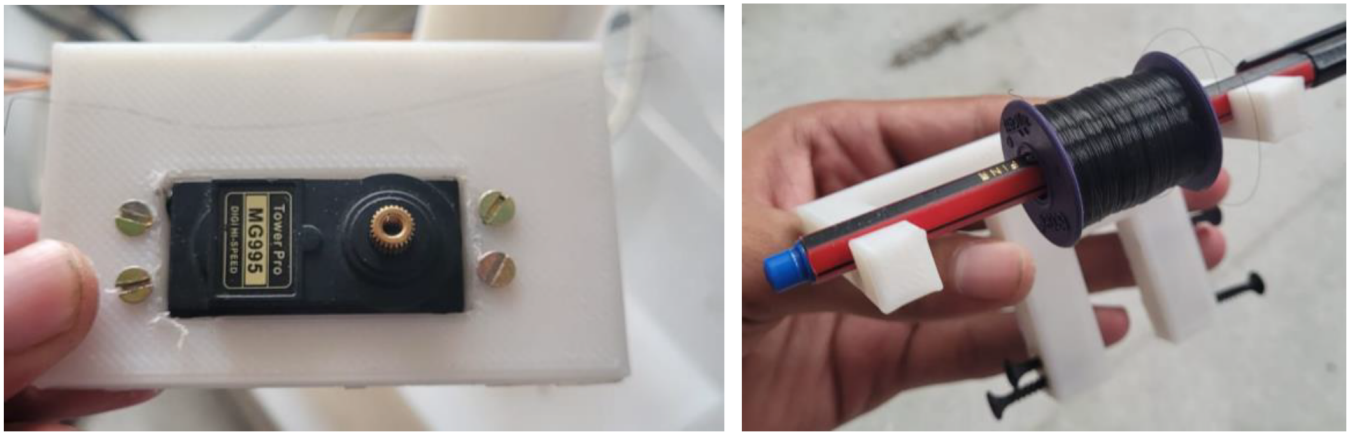
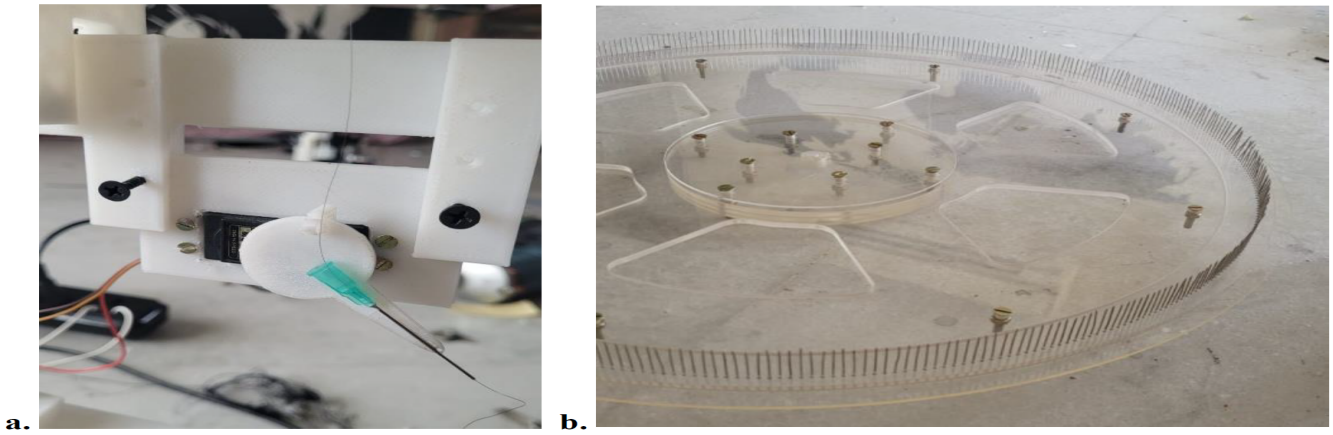


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

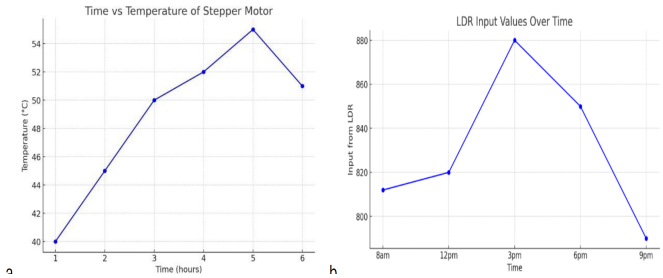


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

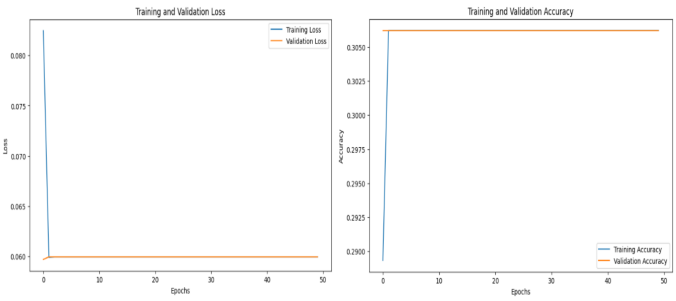


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

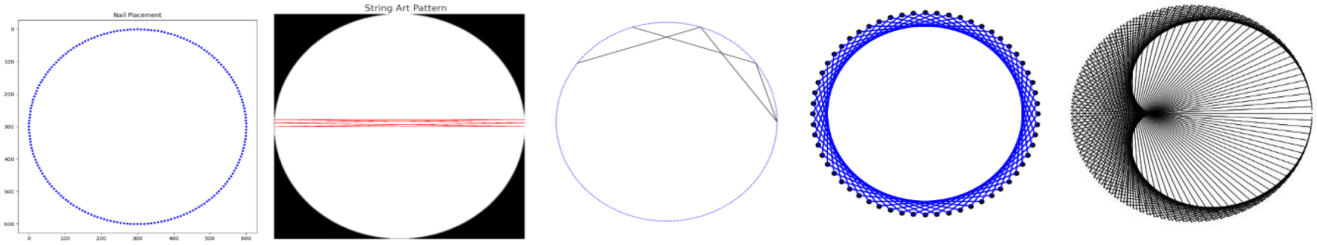


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

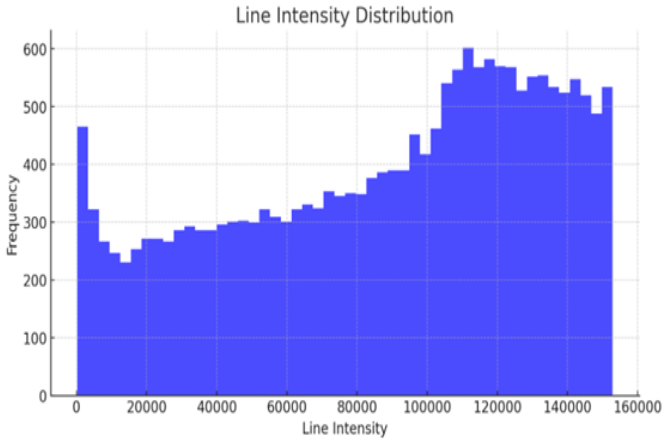


FIGURE 2: Line intensity distribution.

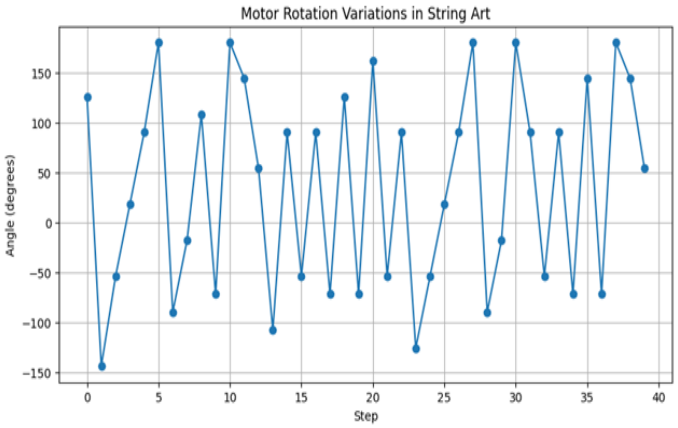


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

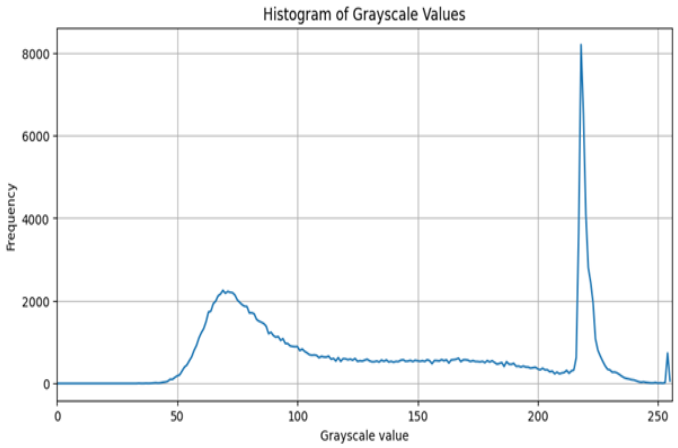


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

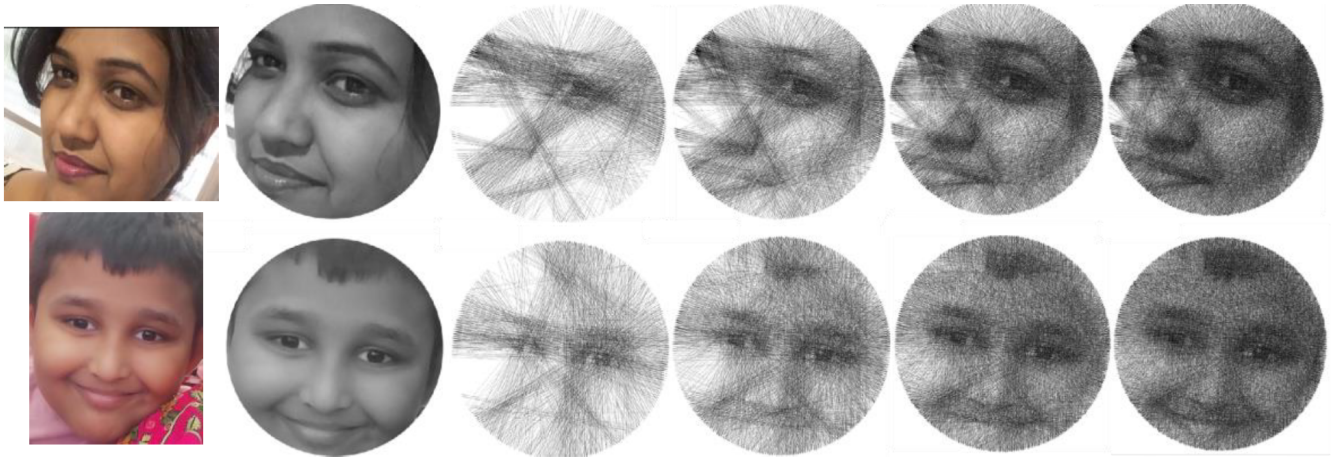


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

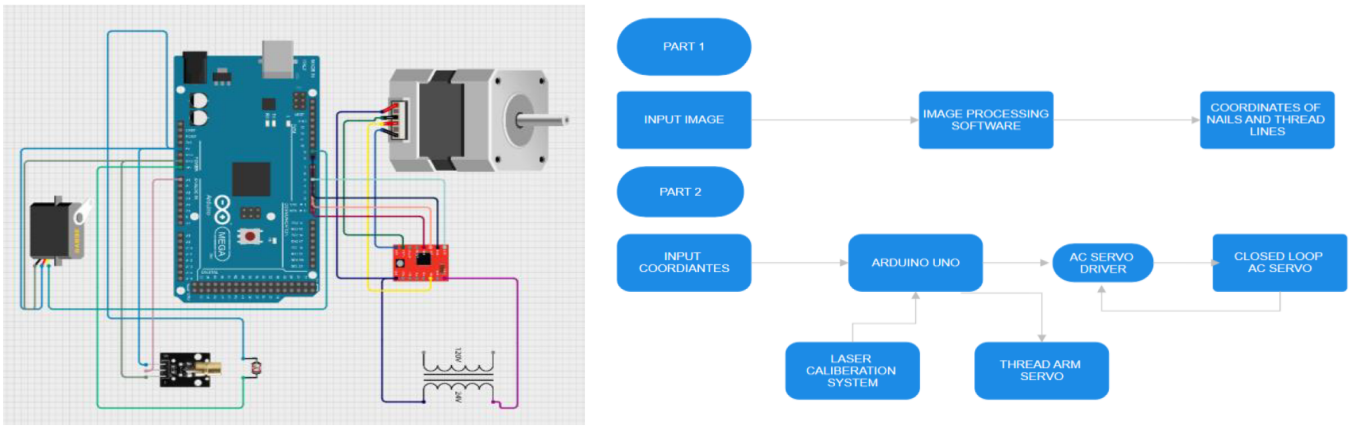


FIGURE 7: Circuit connection with block diagram representation.

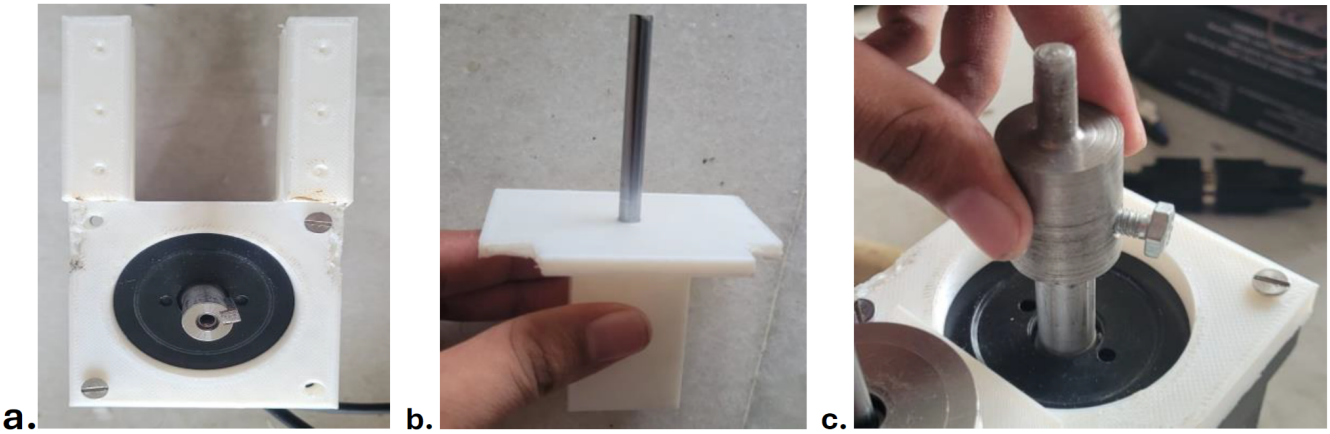


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

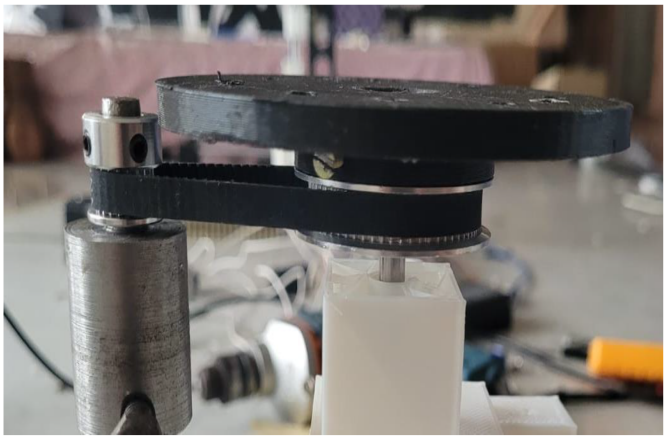


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

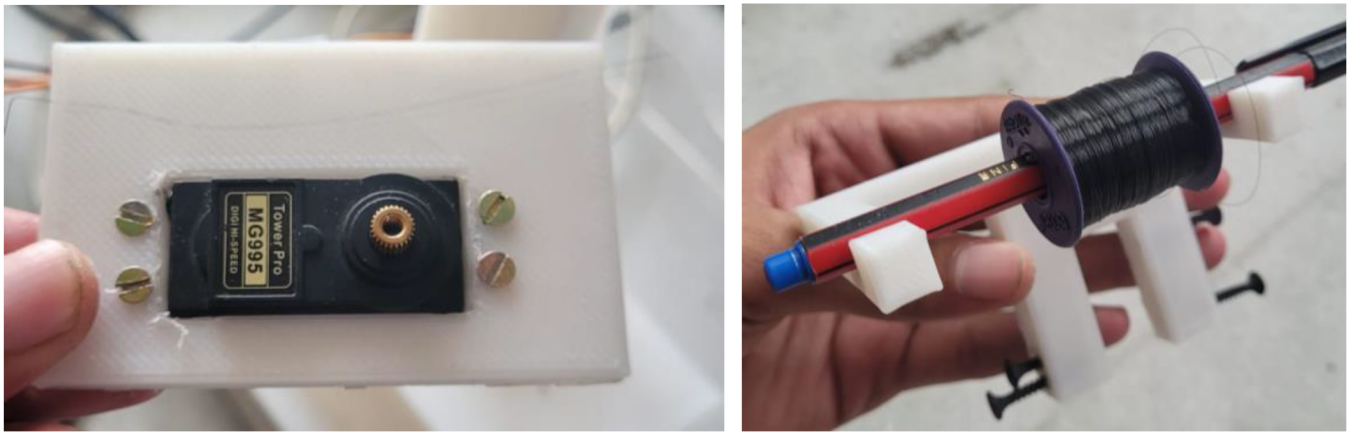
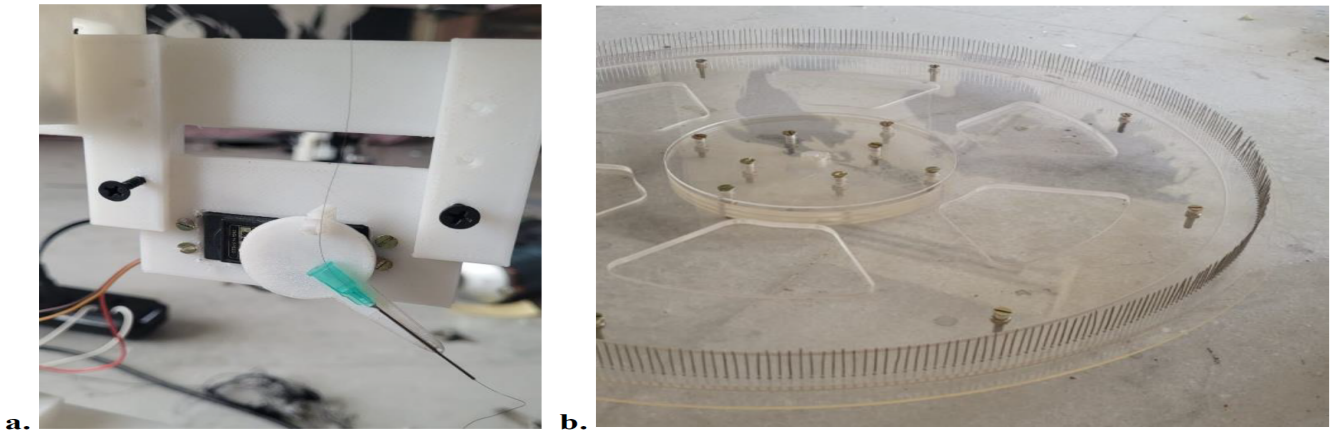


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

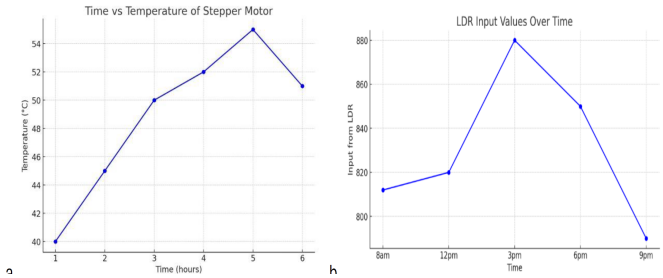


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

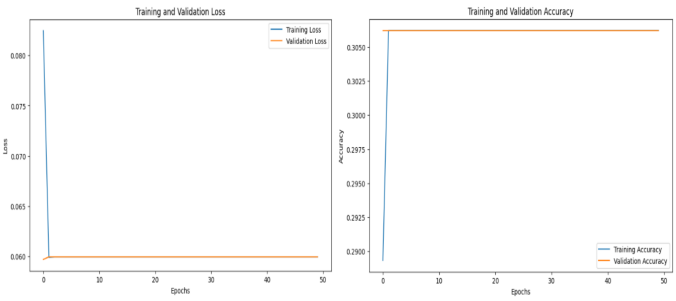


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

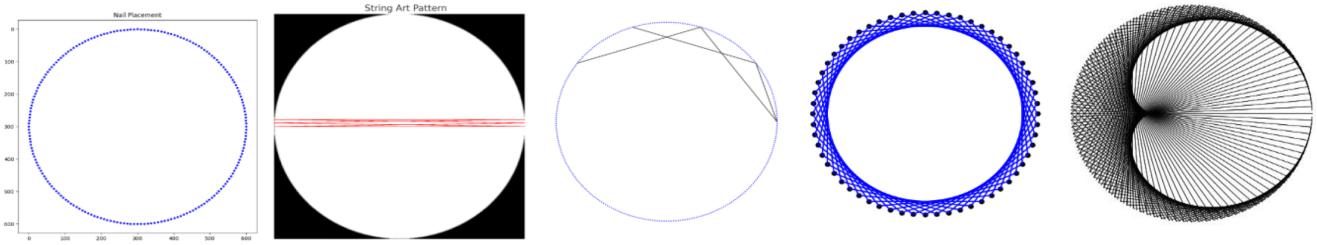


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

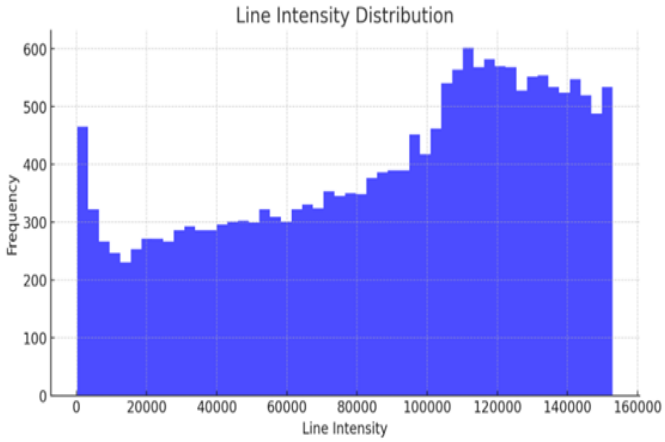


FIGURE 2: Line intensity distribution.

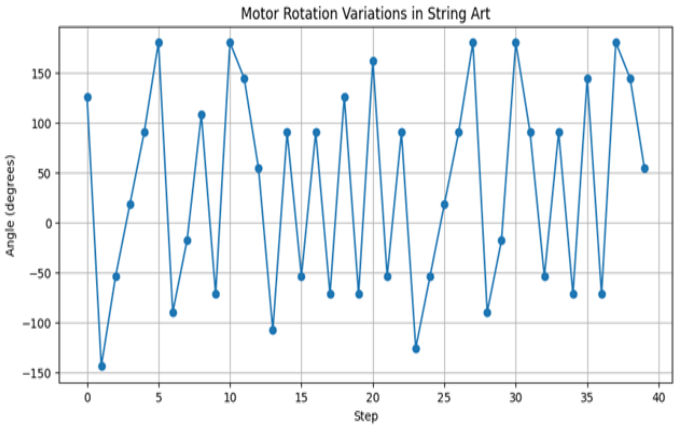


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

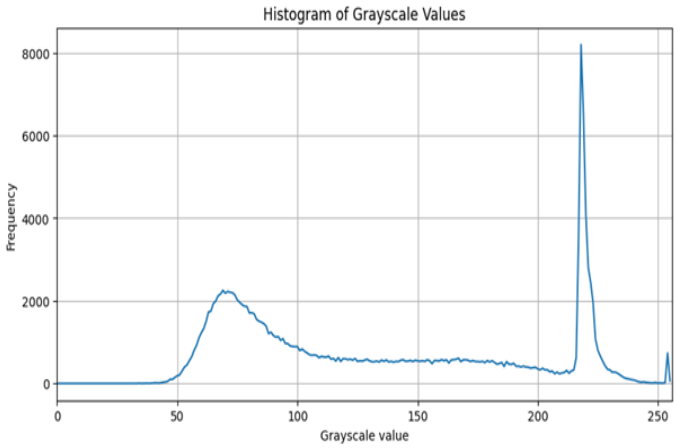


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

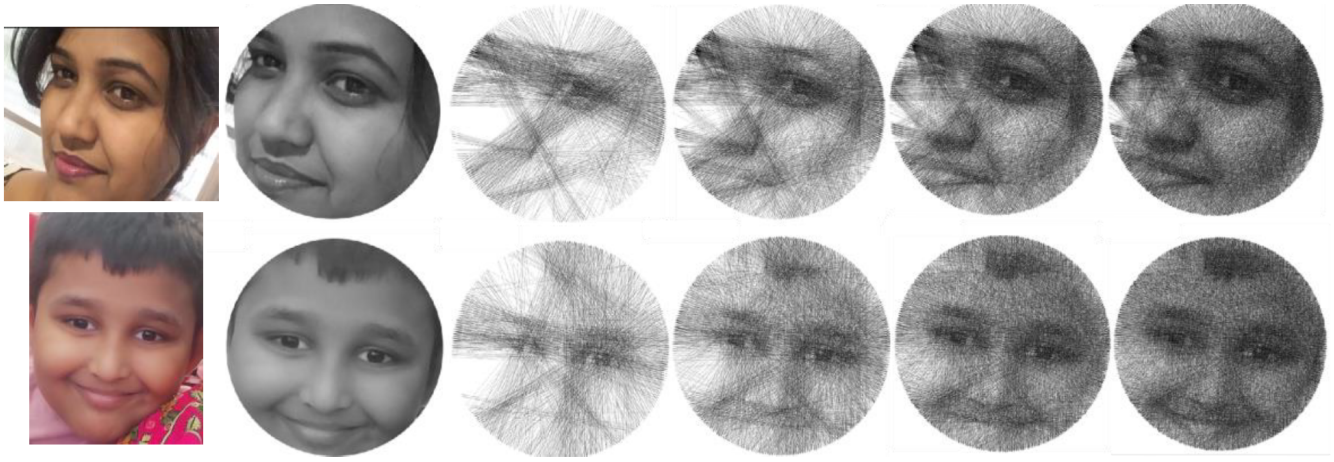


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

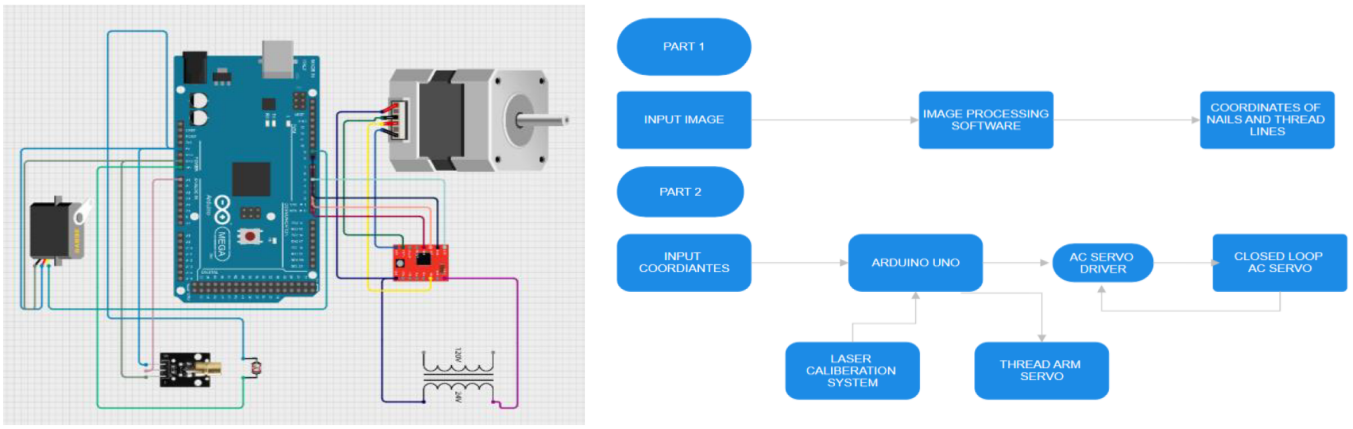


FIGURE 7: Circuit connection with block diagram representation.

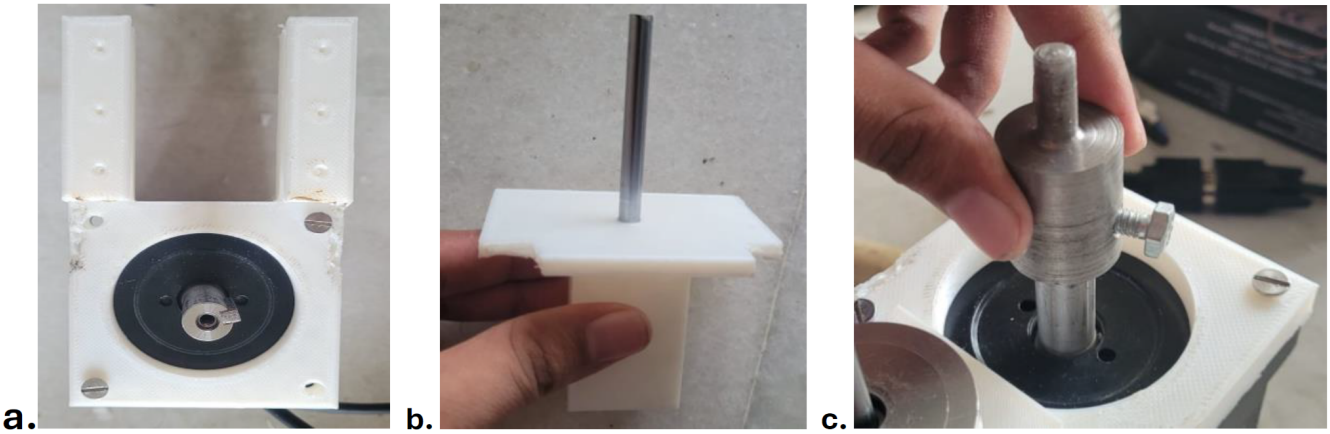


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

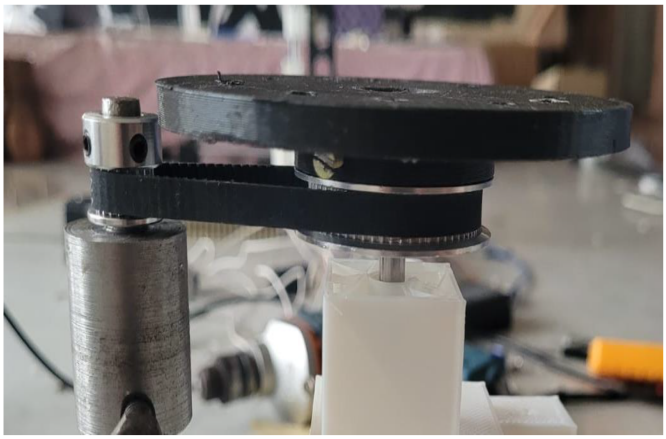


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

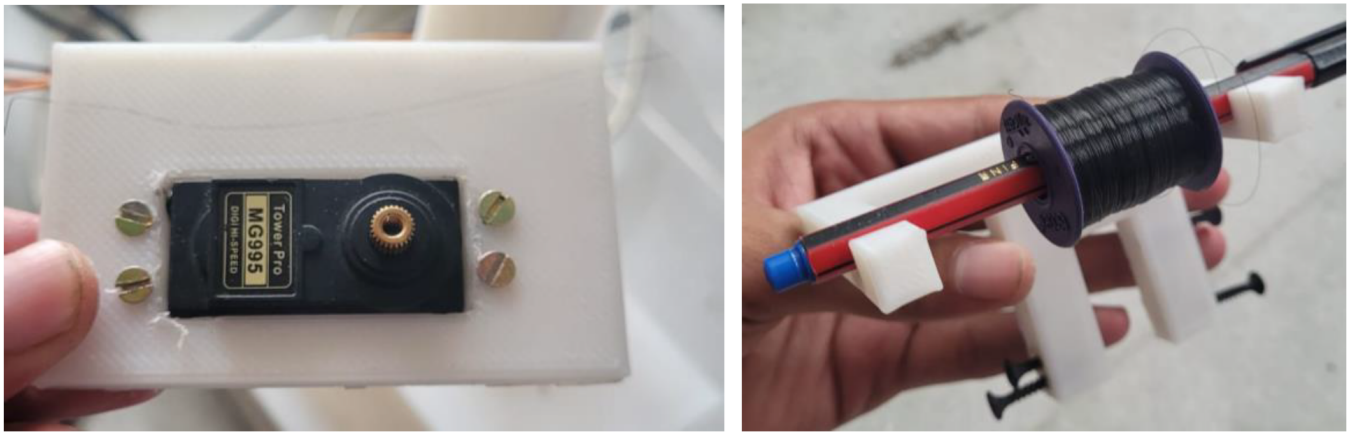
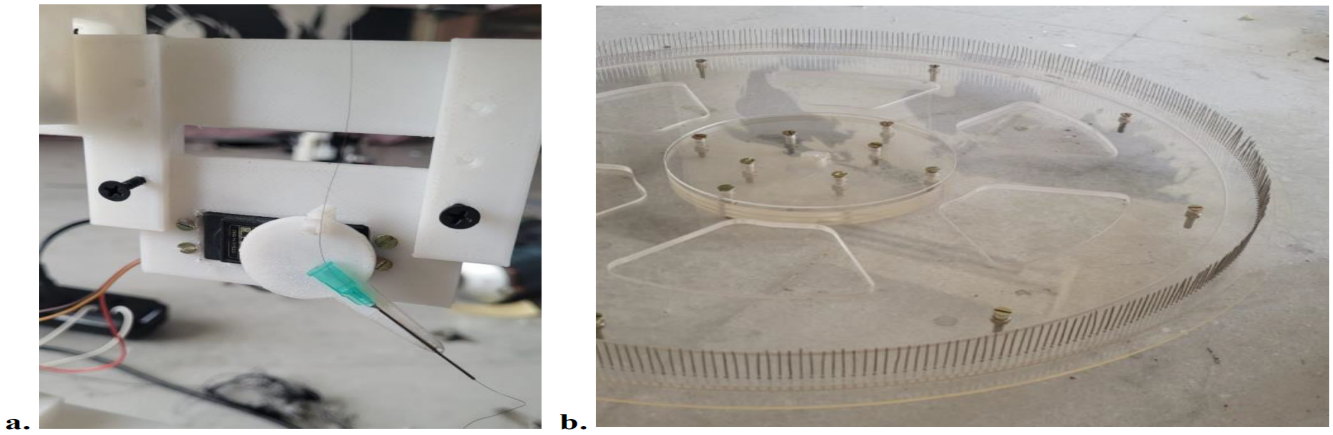


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

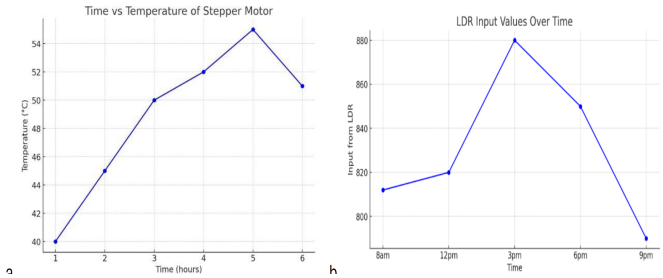


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

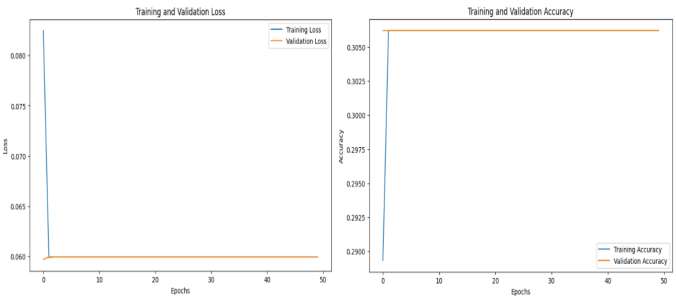


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

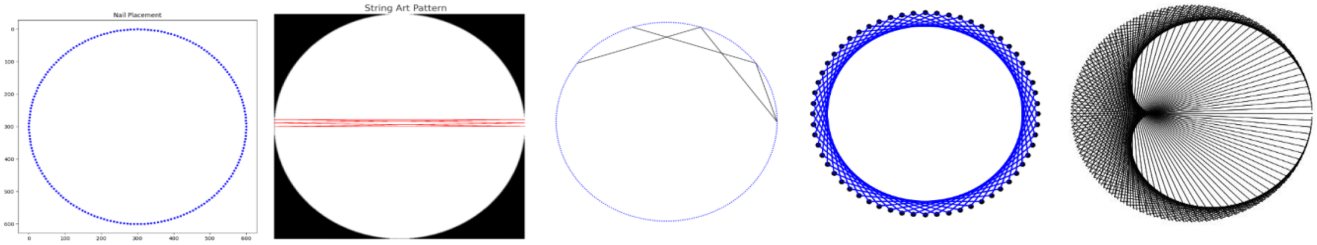


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

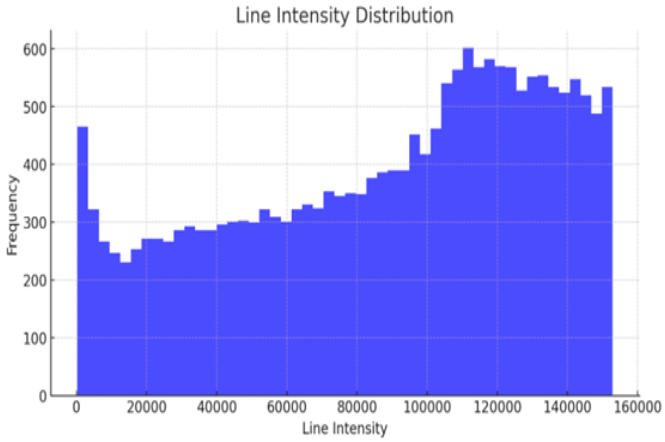


FIGURE 2: Line intensity distribution.

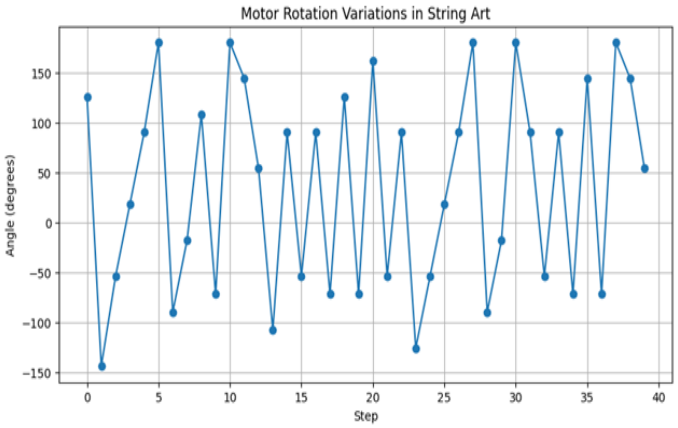


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

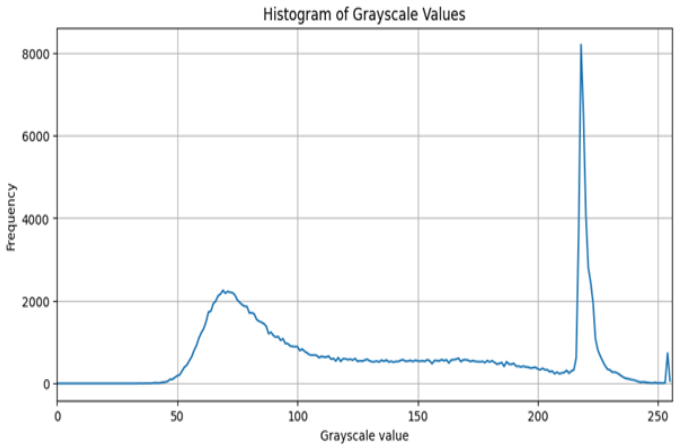


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

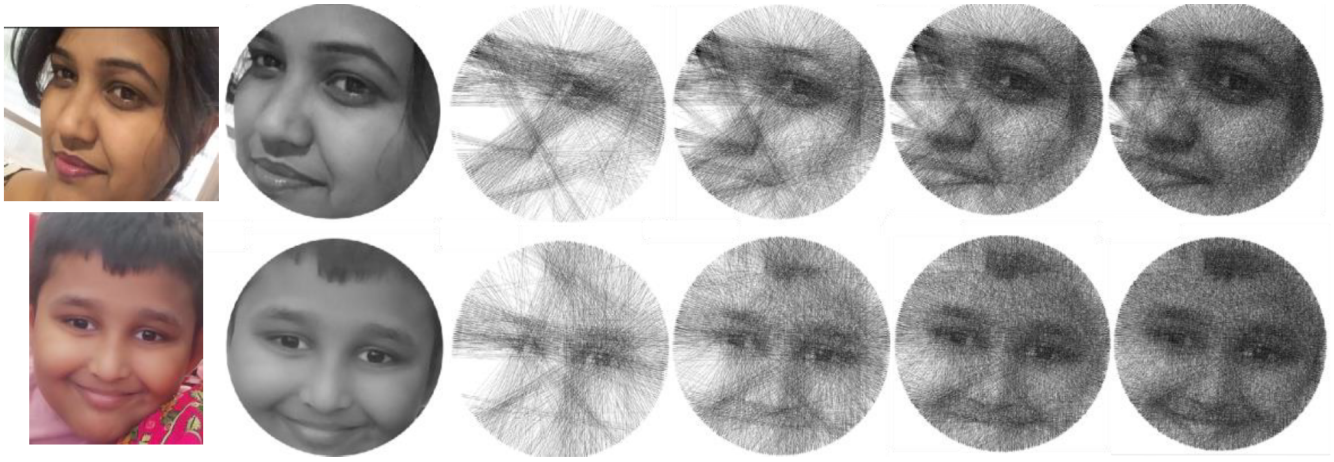


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

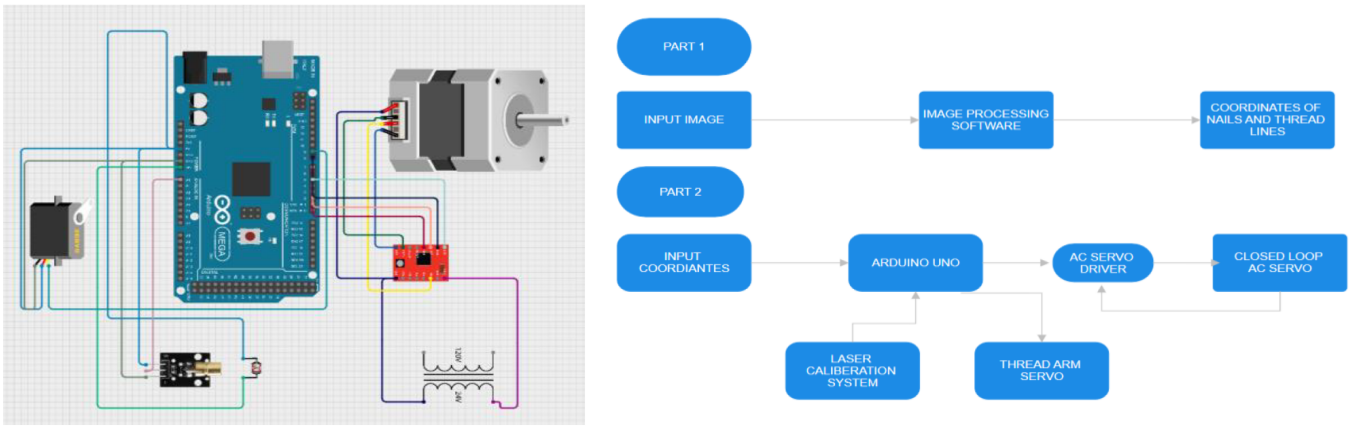


FIGURE 7: Circuit connection with block diagram representation.

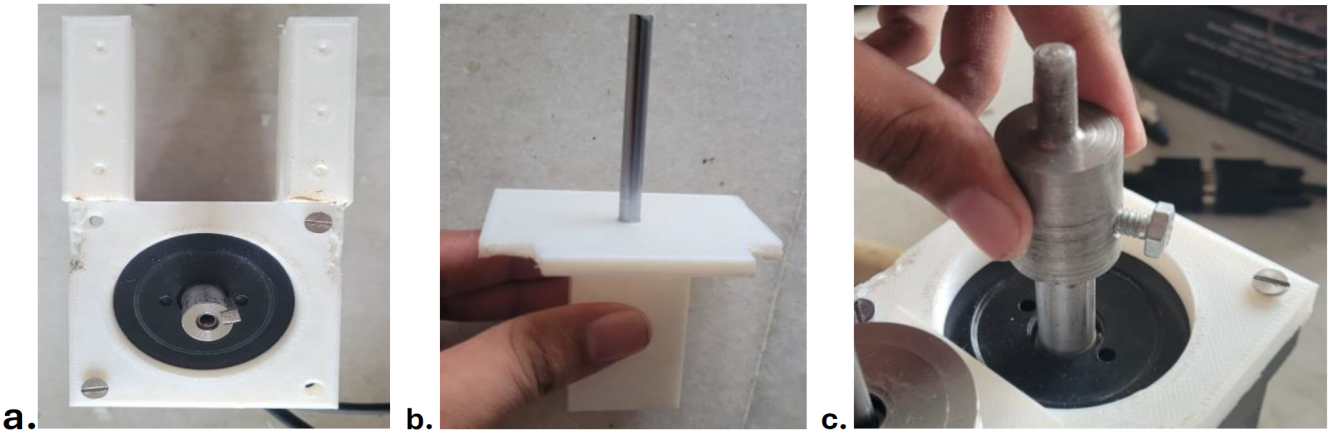


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

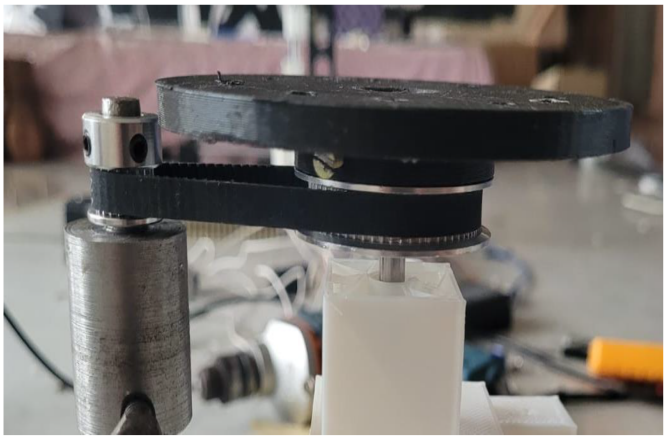


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

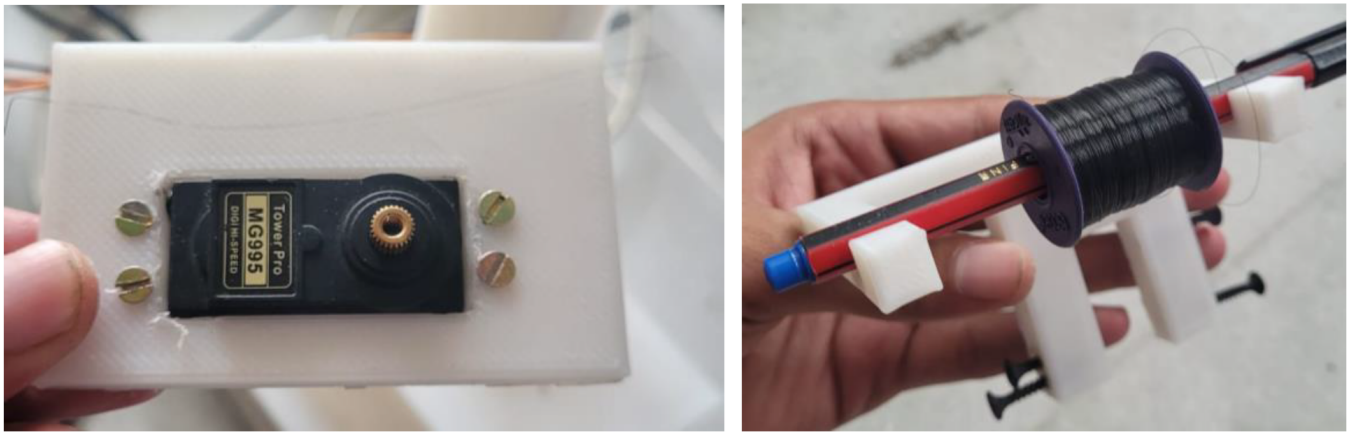
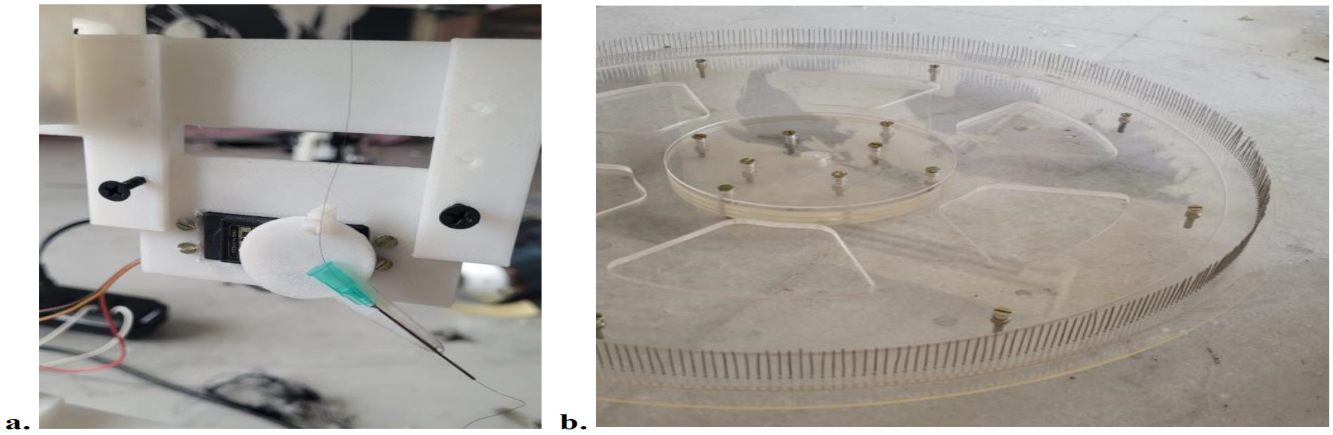


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

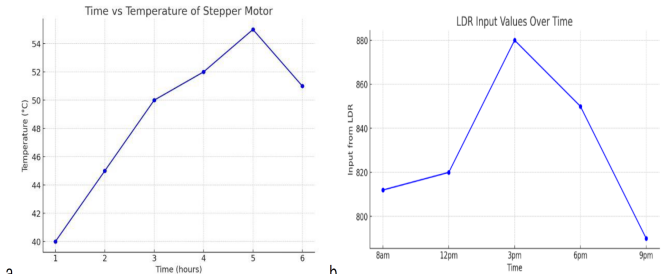


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

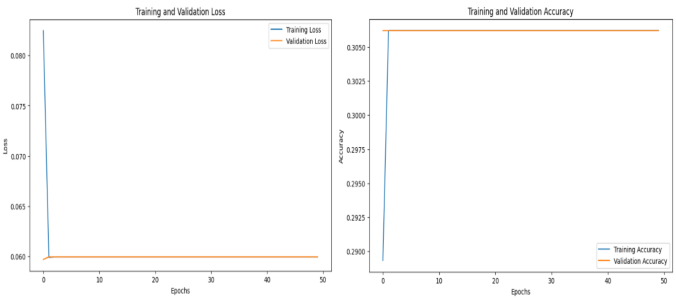


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

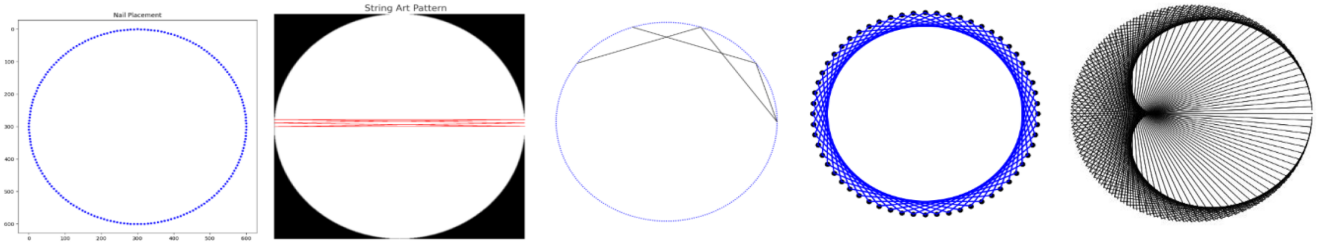


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

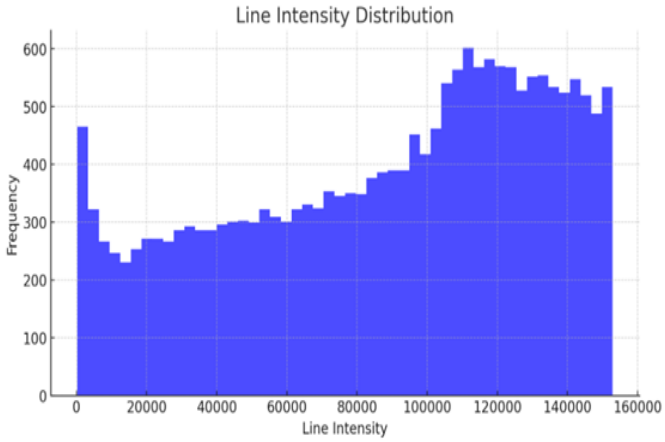


FIGURE 2: Line intensity distribution.

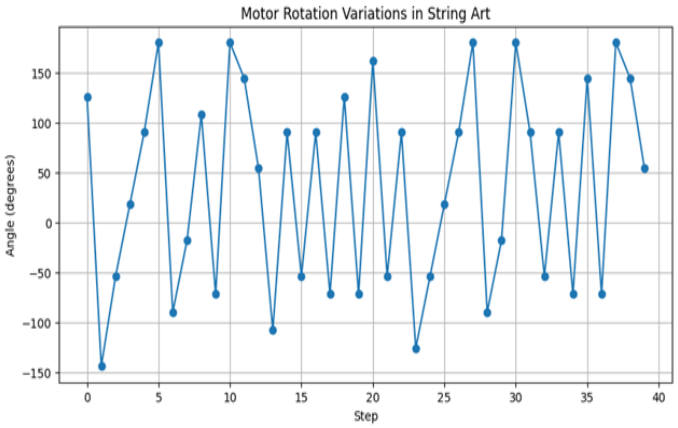


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

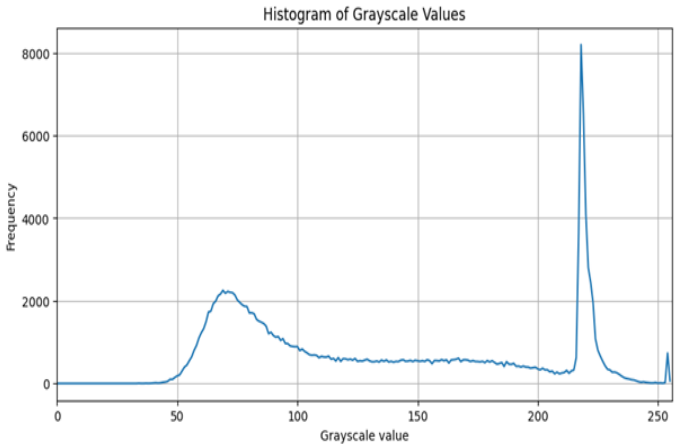


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

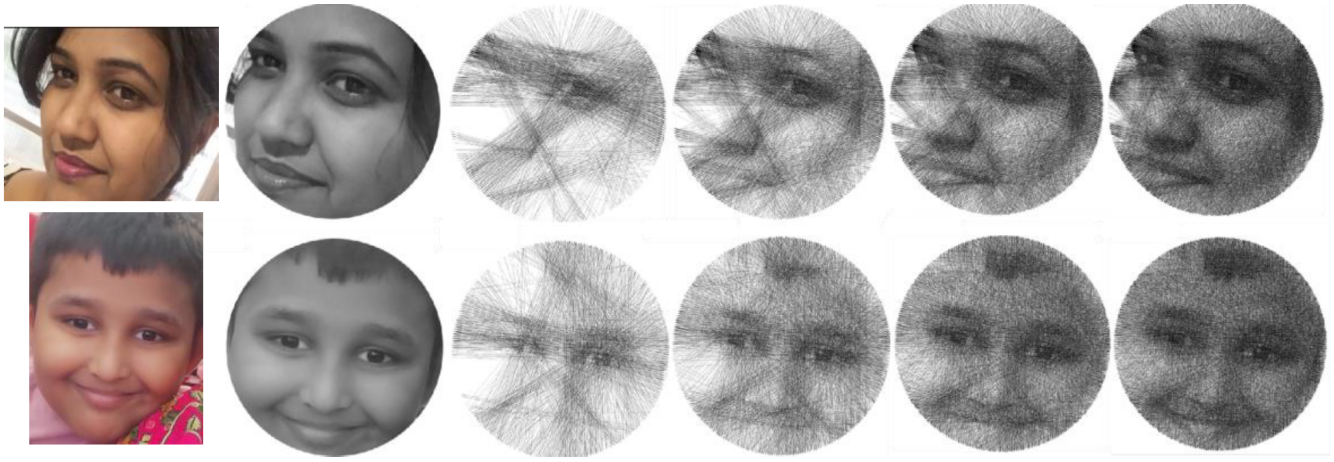


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

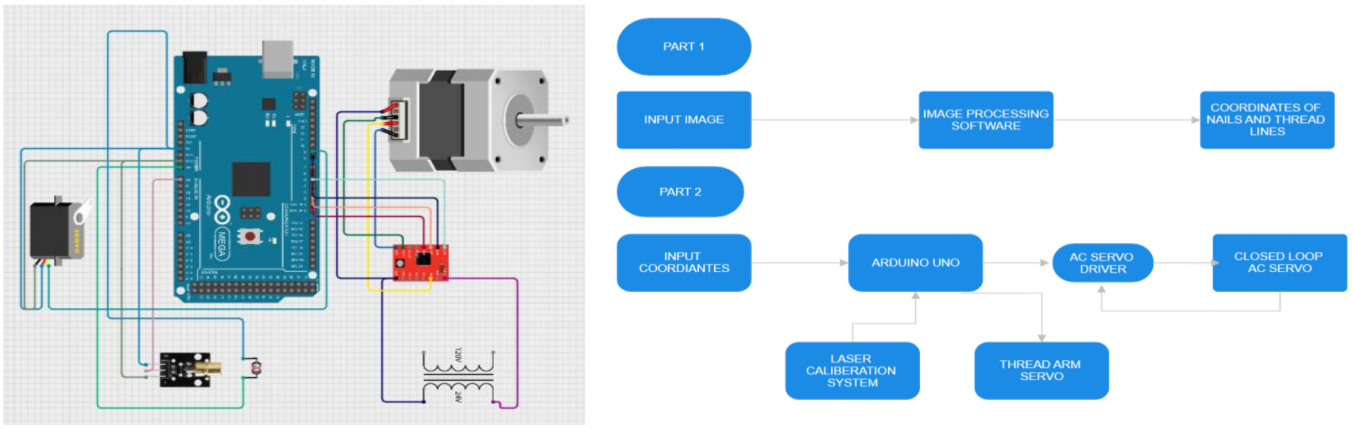


FIGURE 7: Circuit connection with block diagram representation.

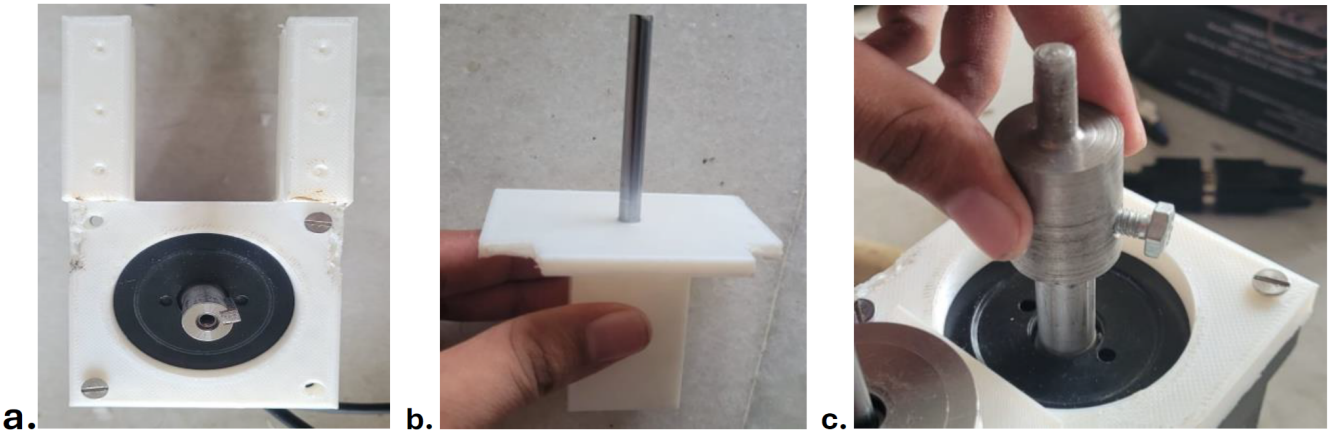


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

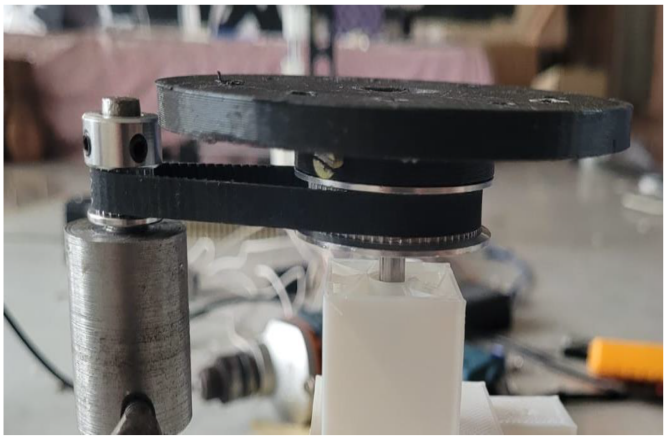


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

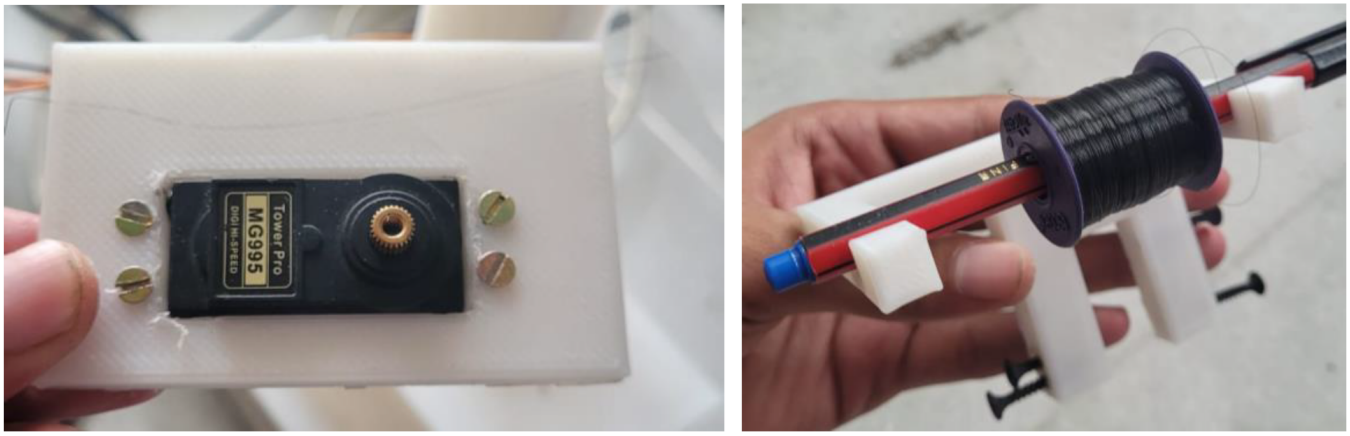
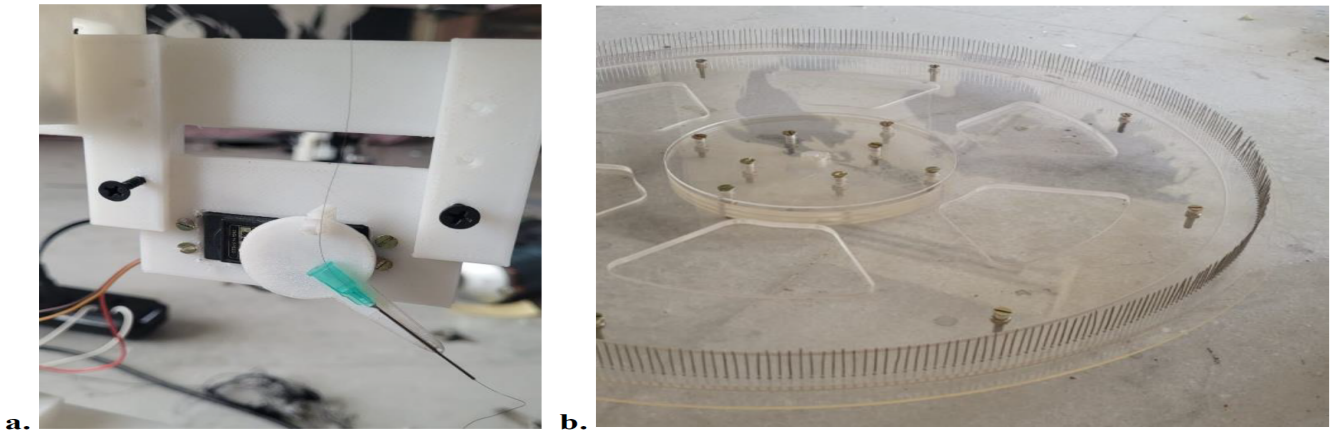


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

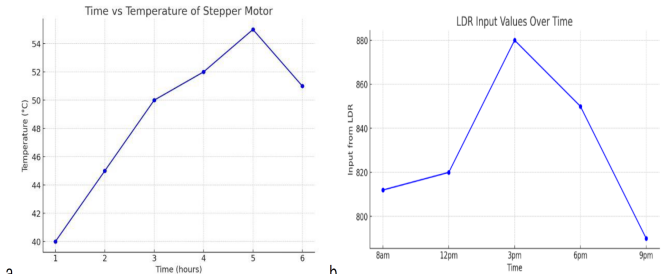


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

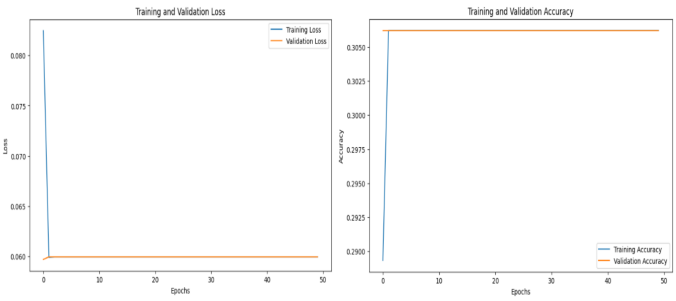


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

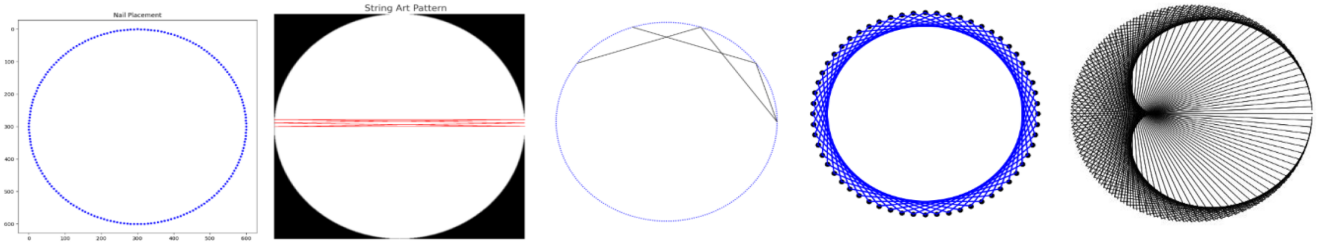


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

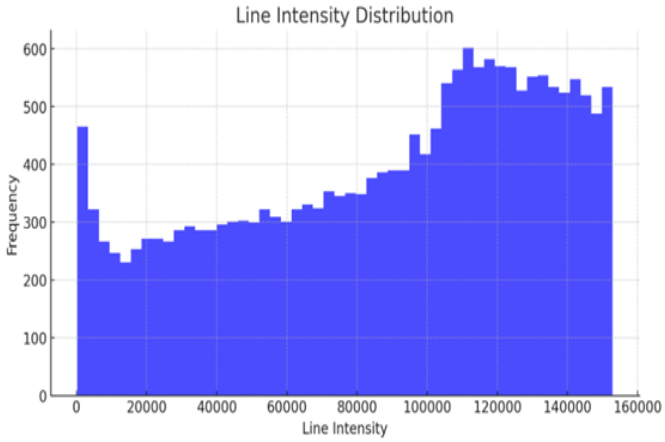


FIGURE 2: Line intensity distribution.

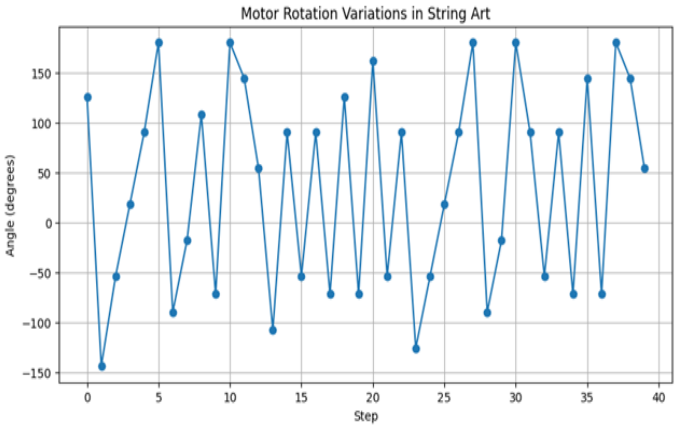


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

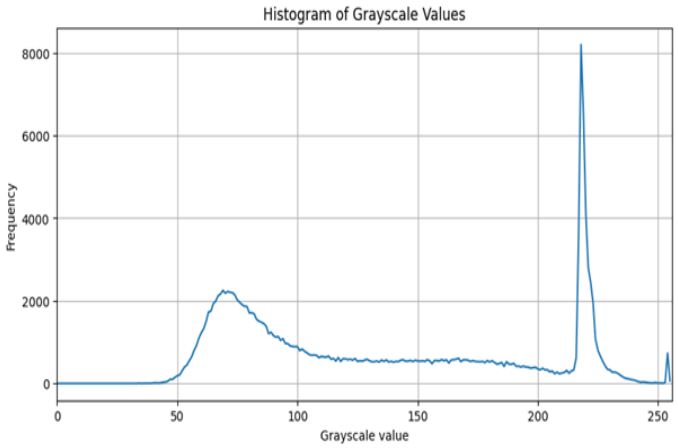


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

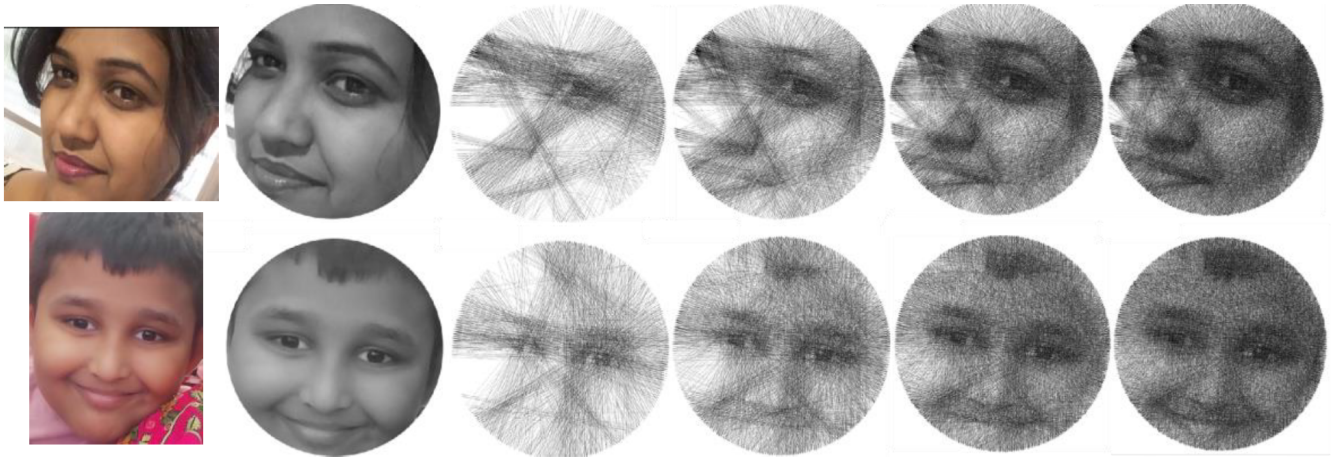


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

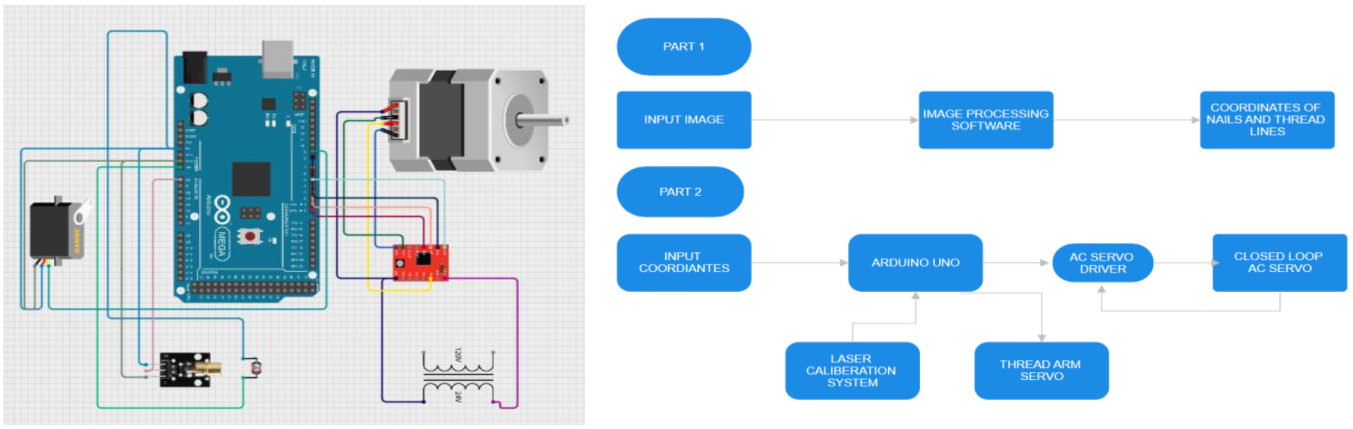


FIGURE 7: Circuit connection with block diagram representation.

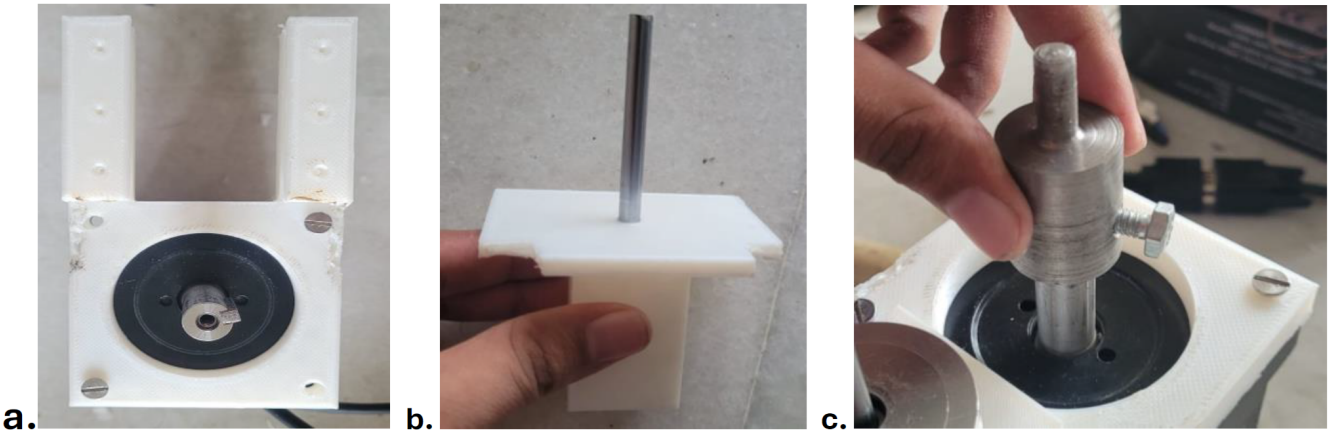


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

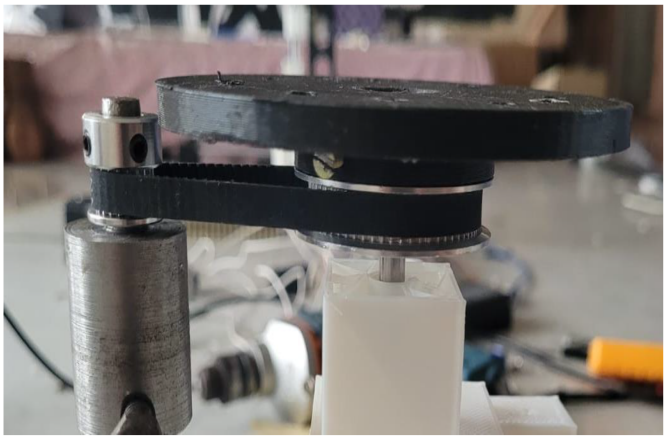


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

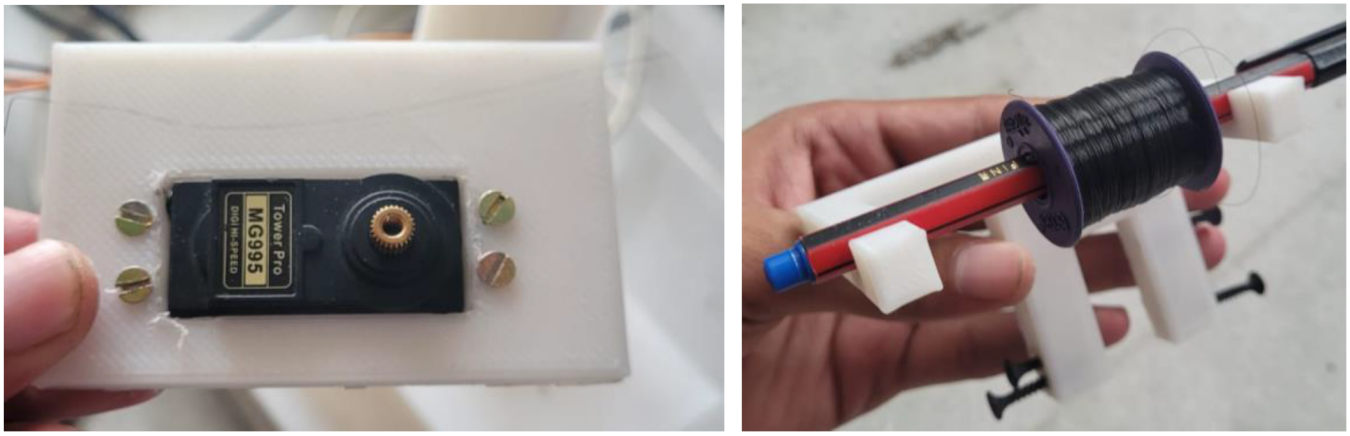
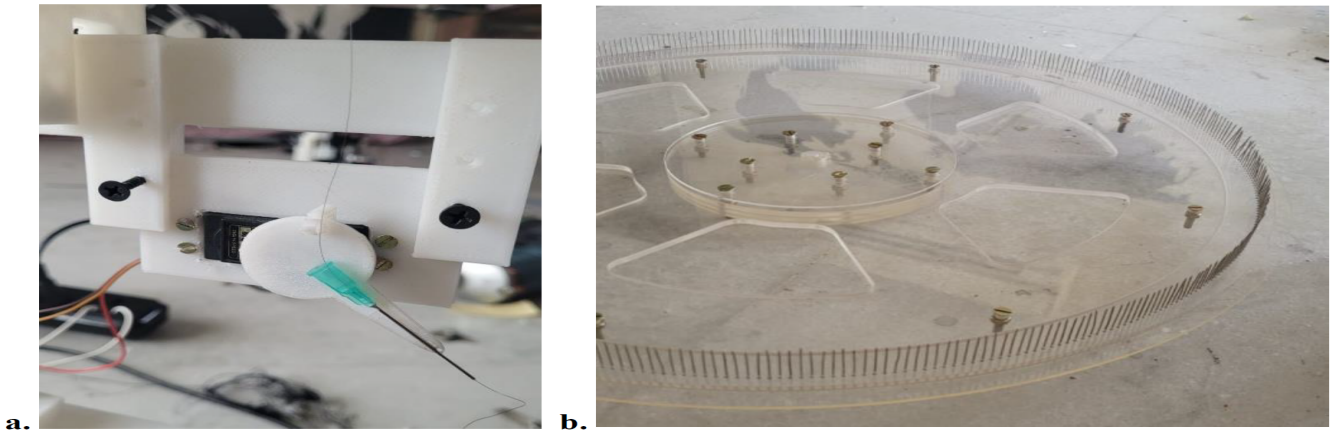


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

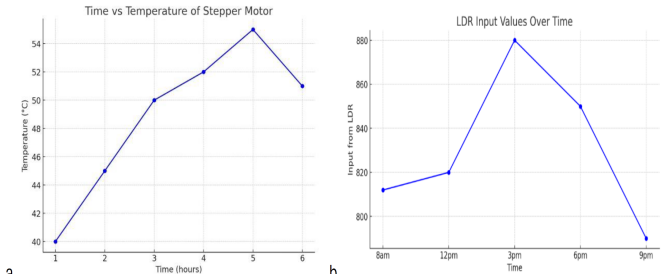


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

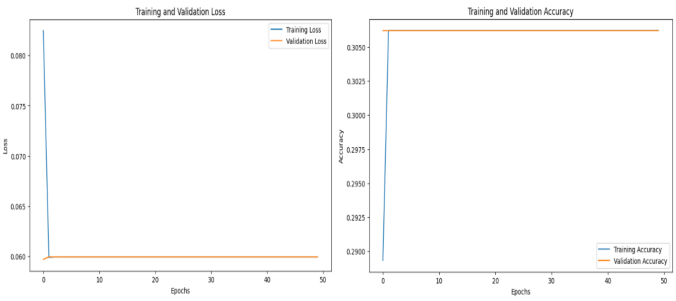


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

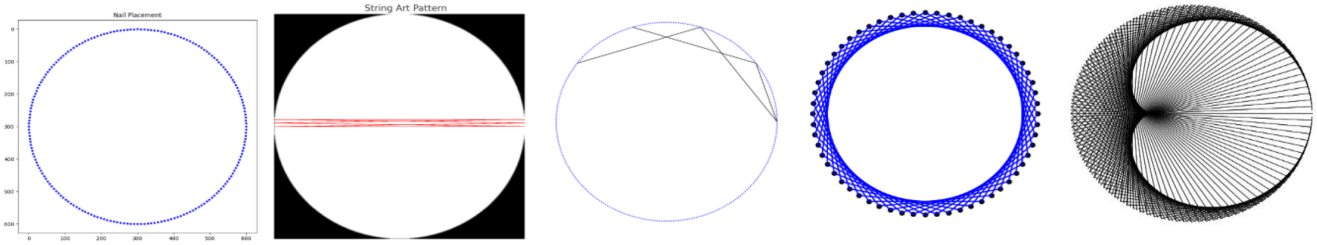


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

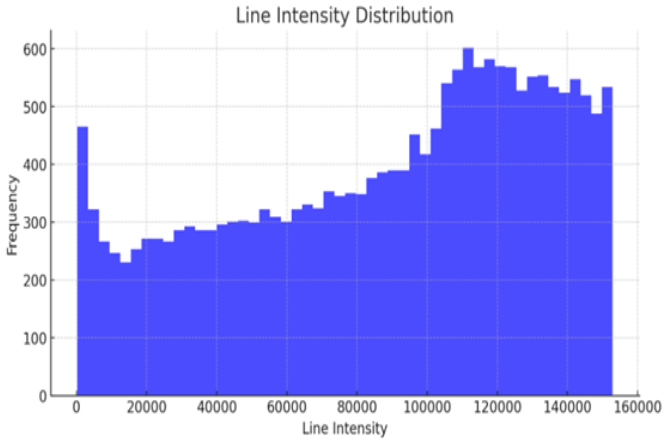


FIGURE 2: Line intensity distribution.

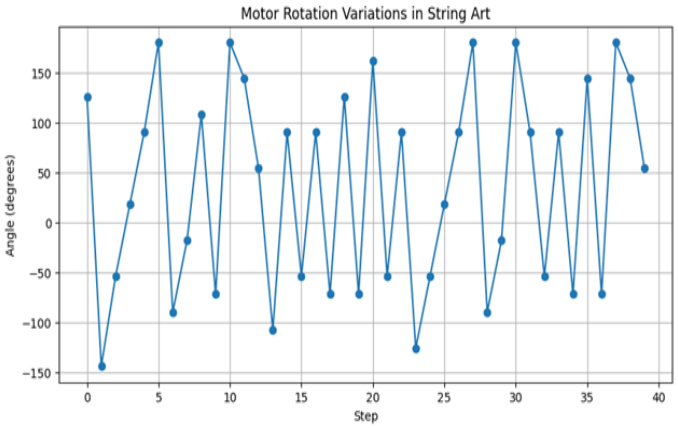


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

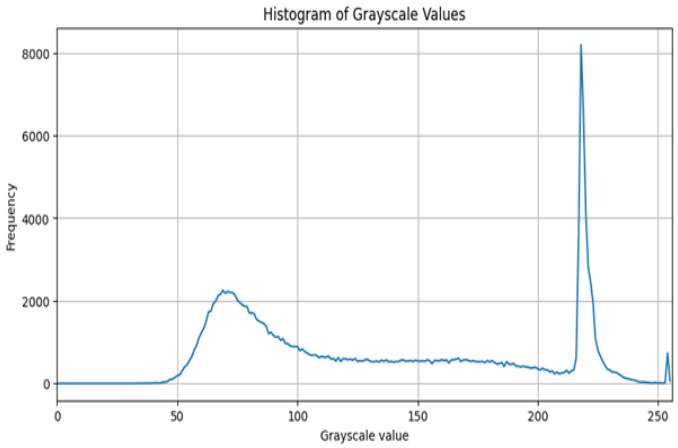


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

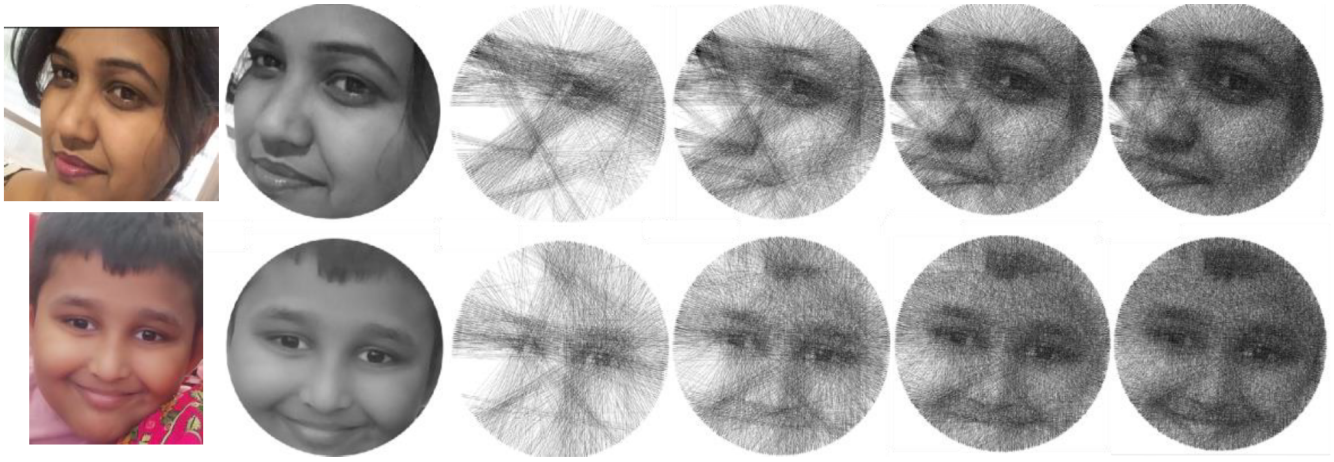


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

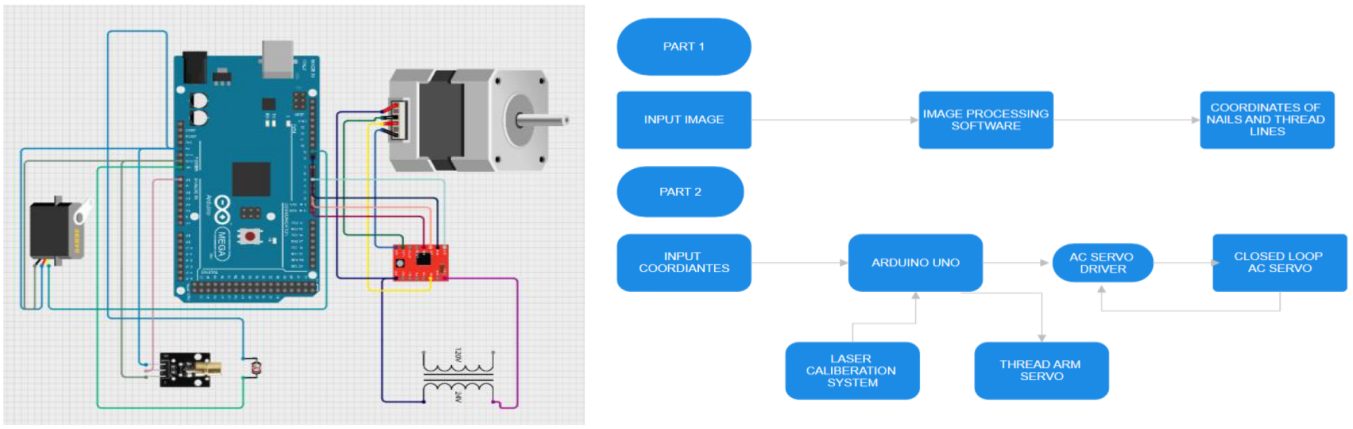


FIGURE 7: Circuit connection with block diagram representation.

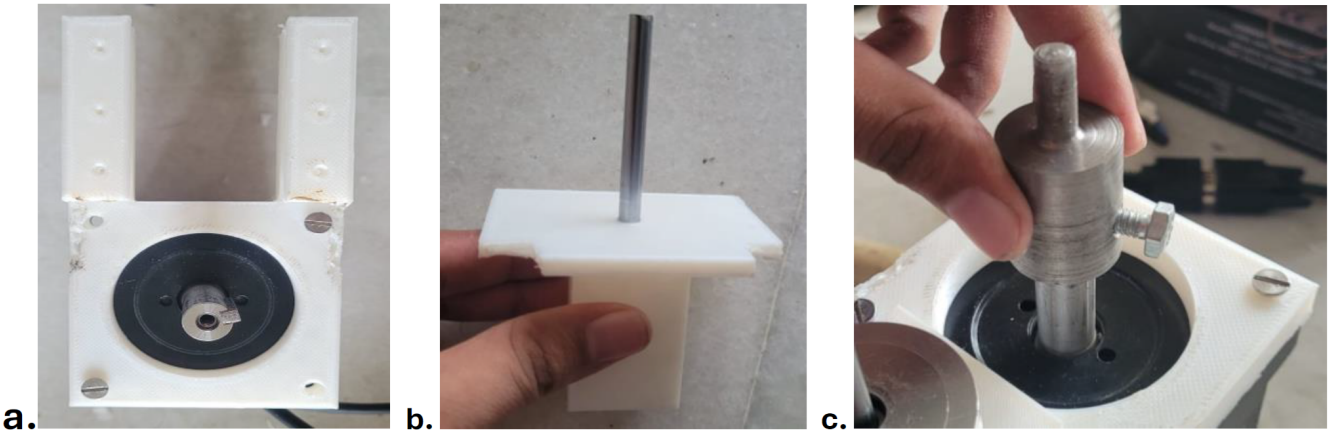


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

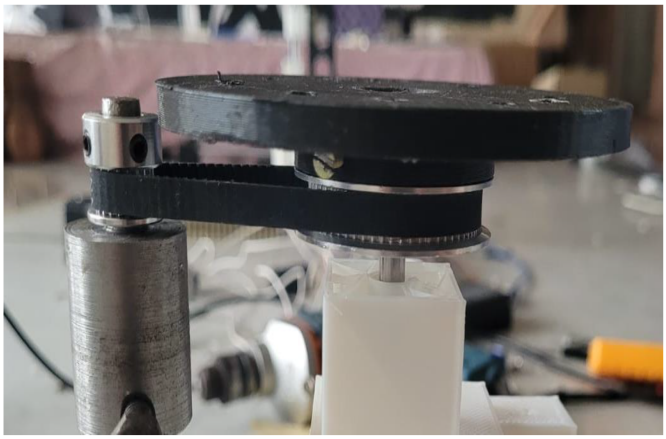


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

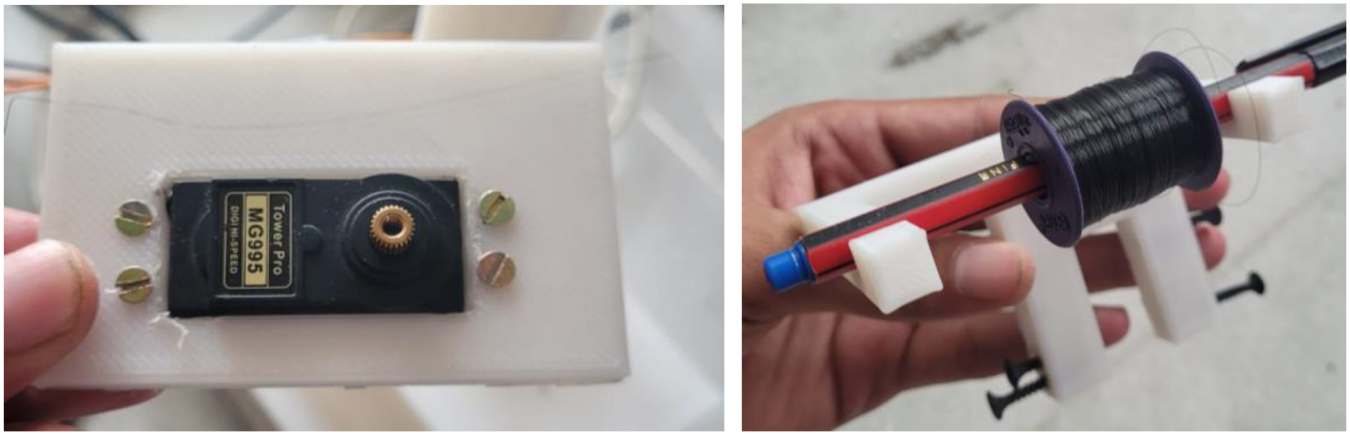
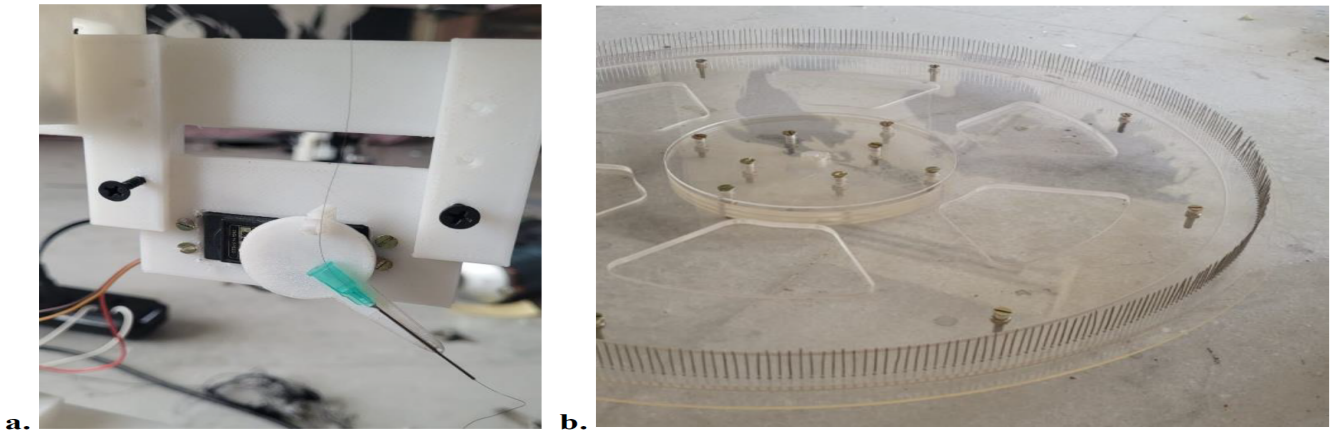


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

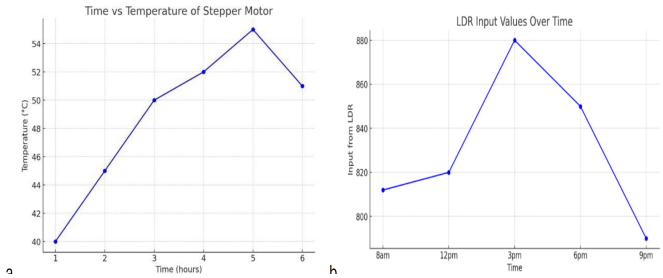


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

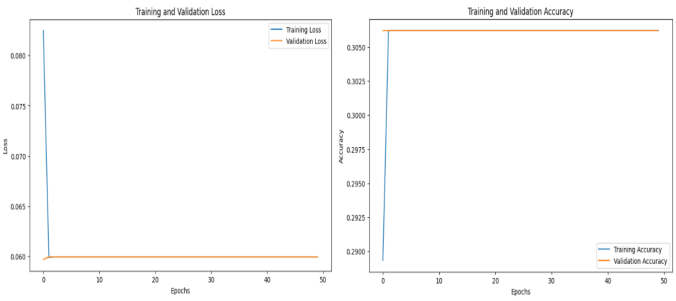


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

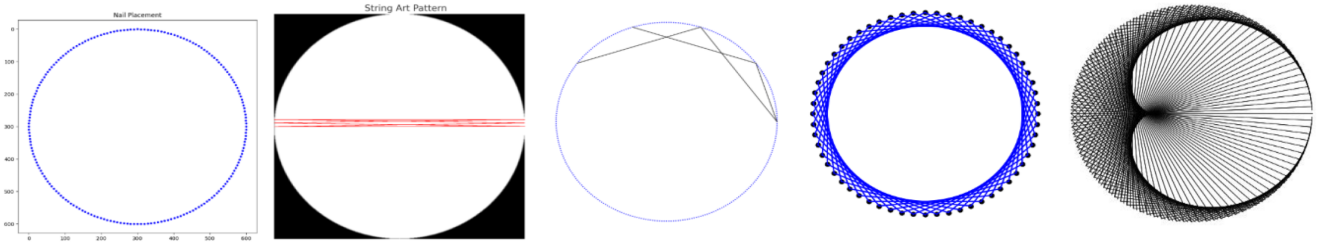


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

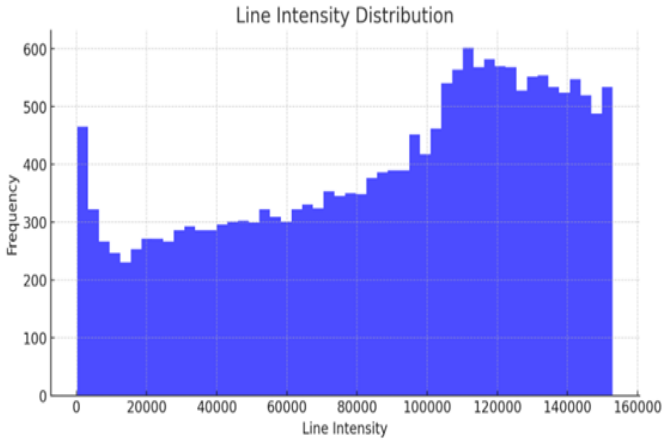


FIGURE 2: Line intensity distribution.

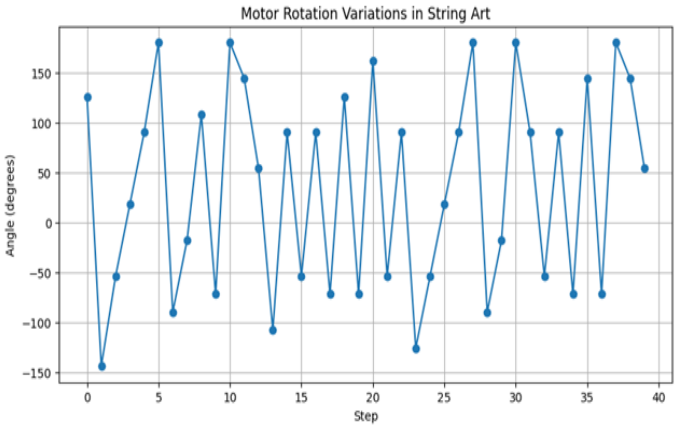


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

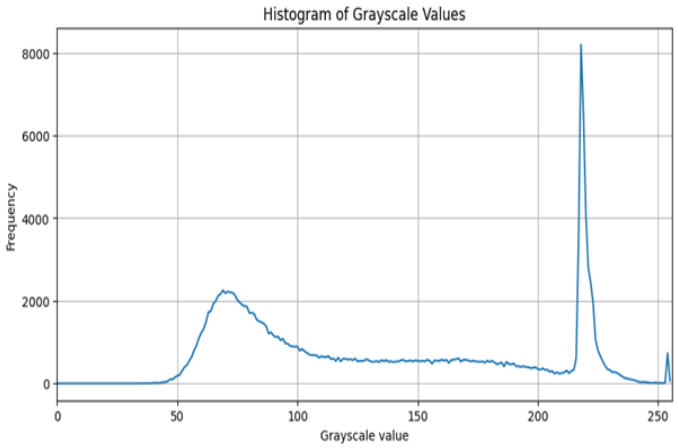


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

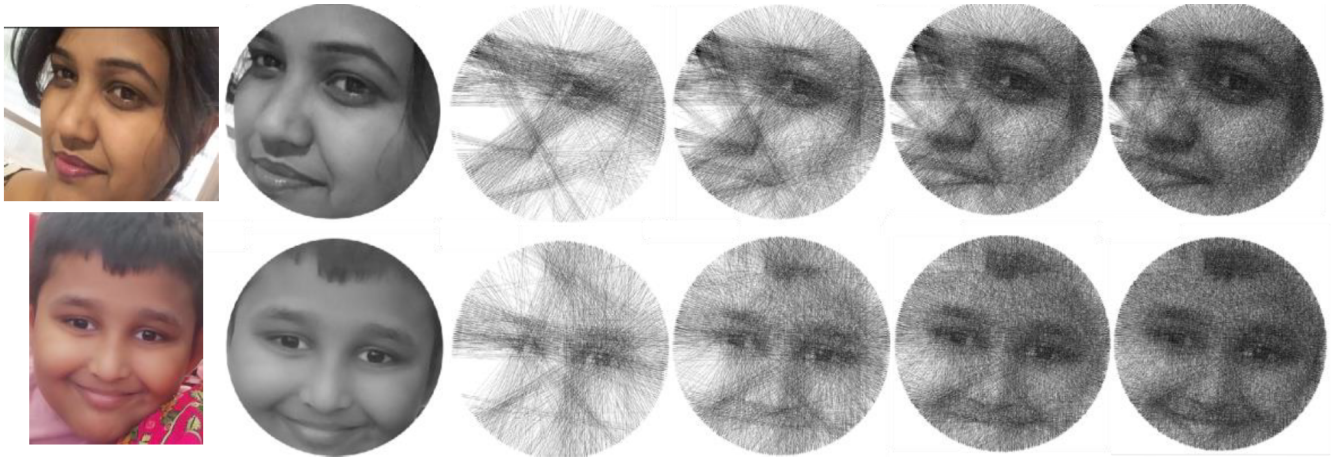


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

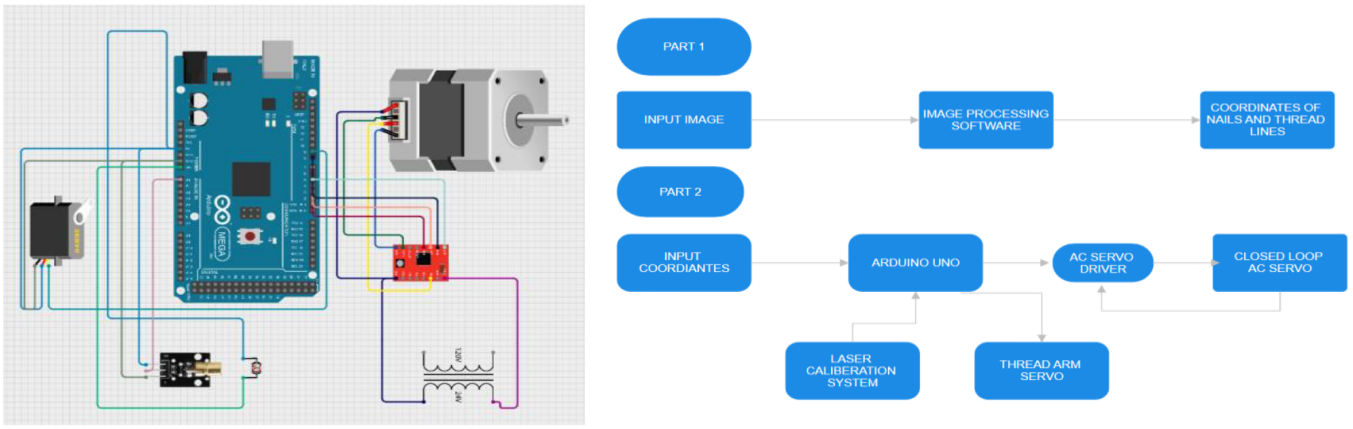


FIGURE 7: Circuit connection with block diagram representation.

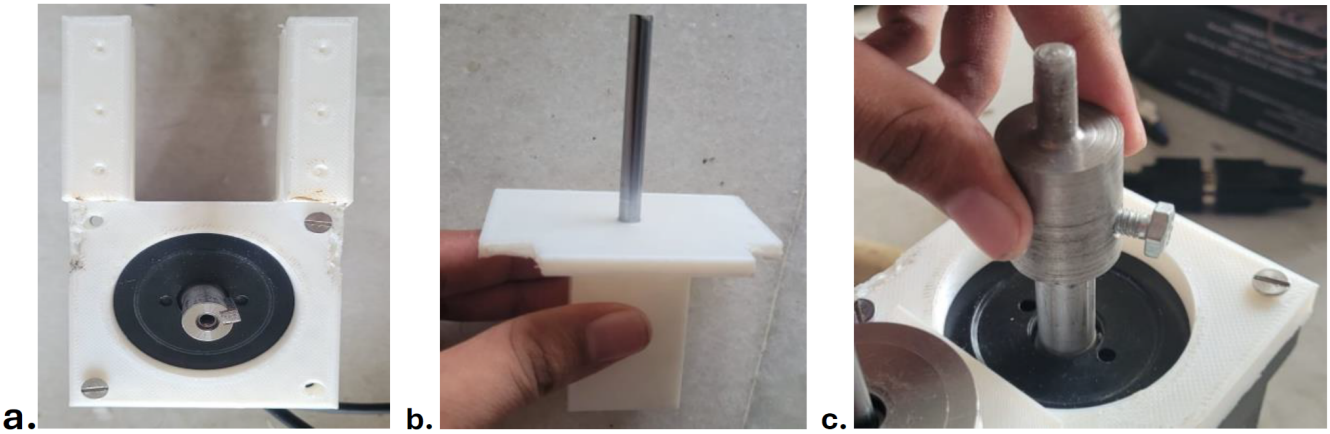


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

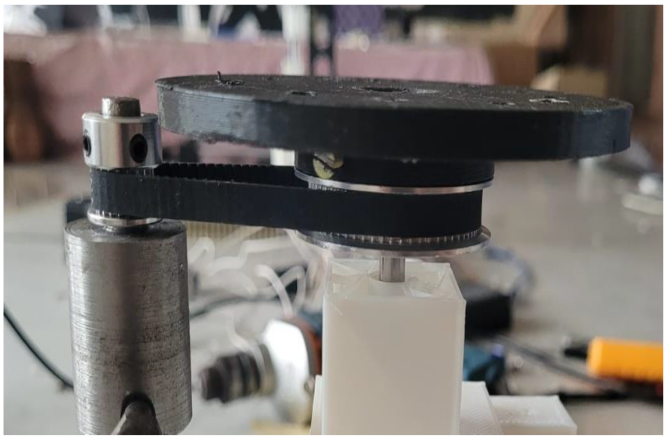


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

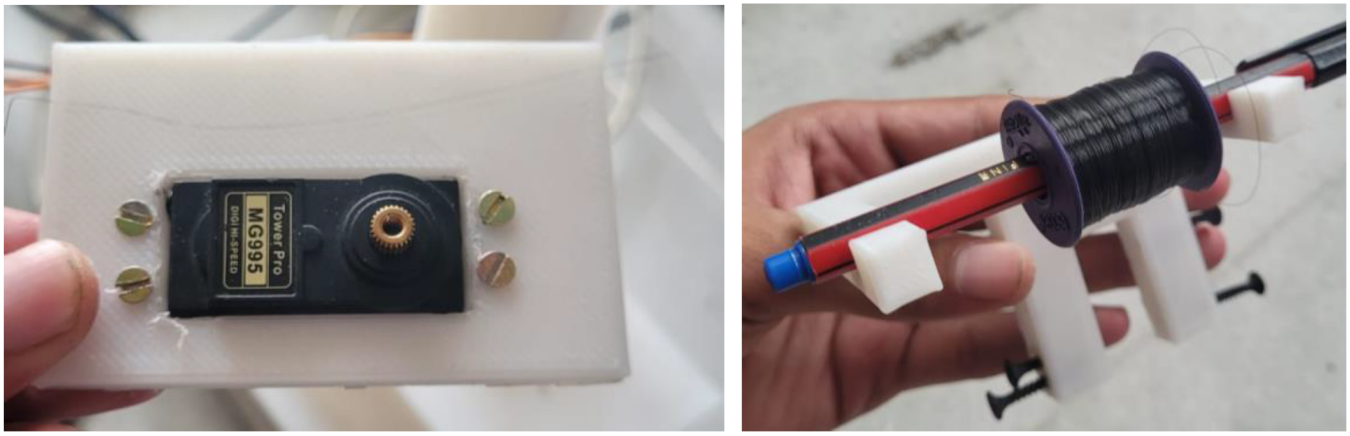
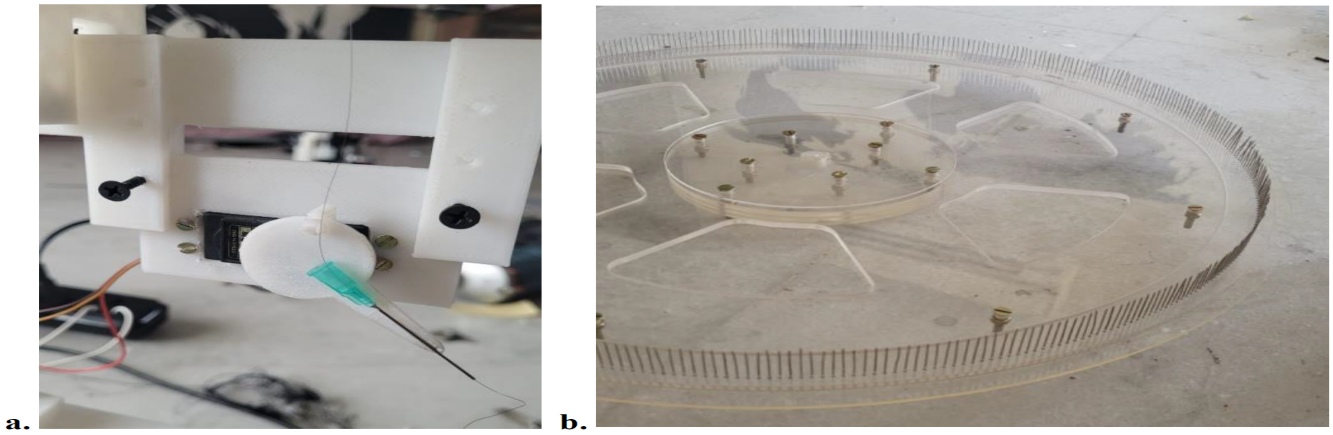


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

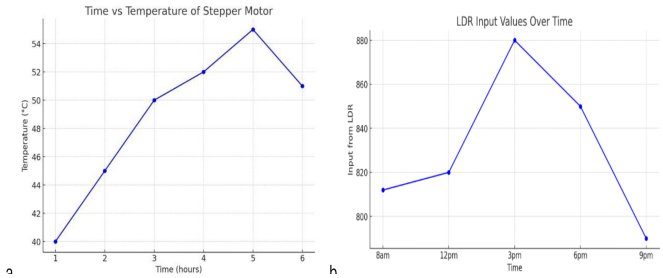


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

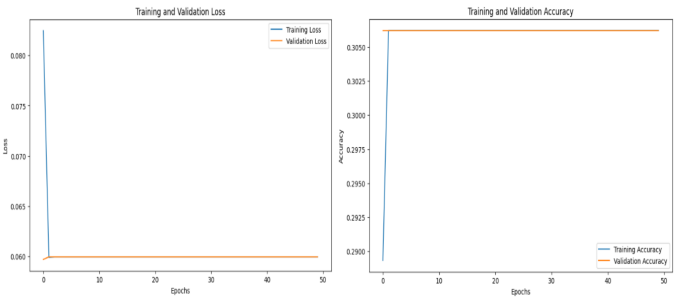


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

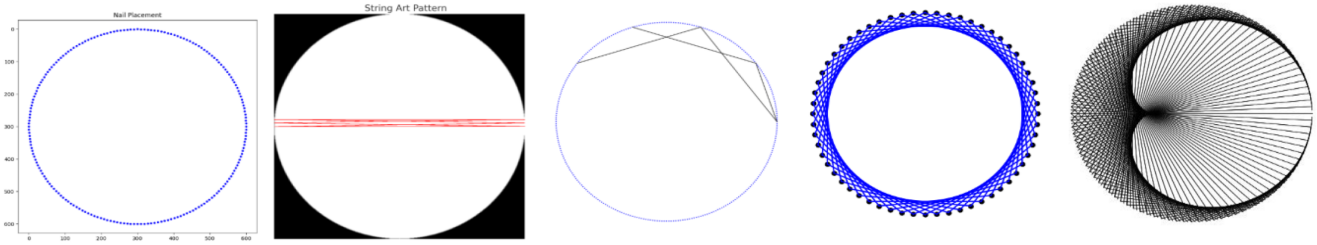


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

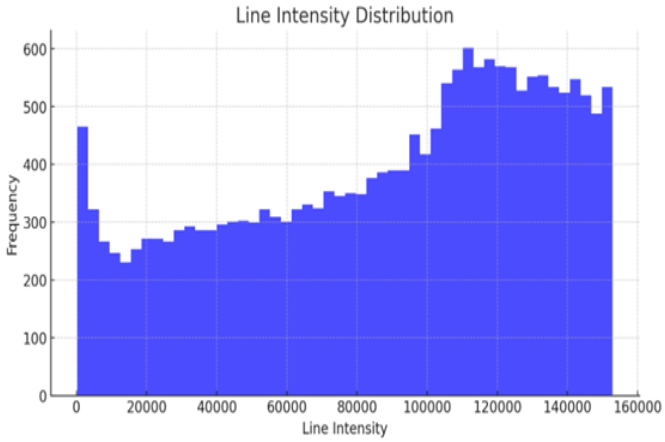


FIGURE 2: Line intensity distribution.

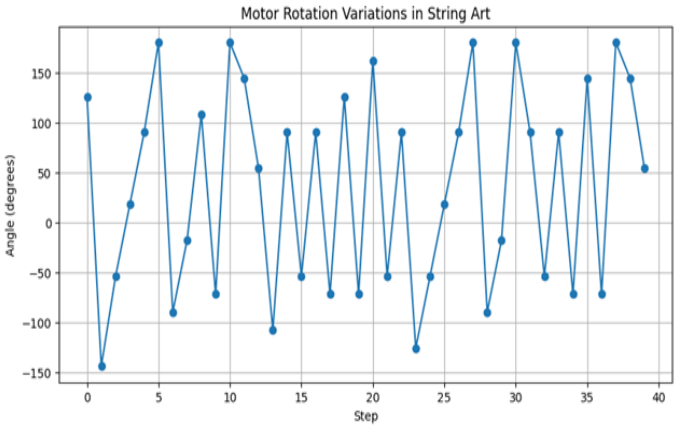


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

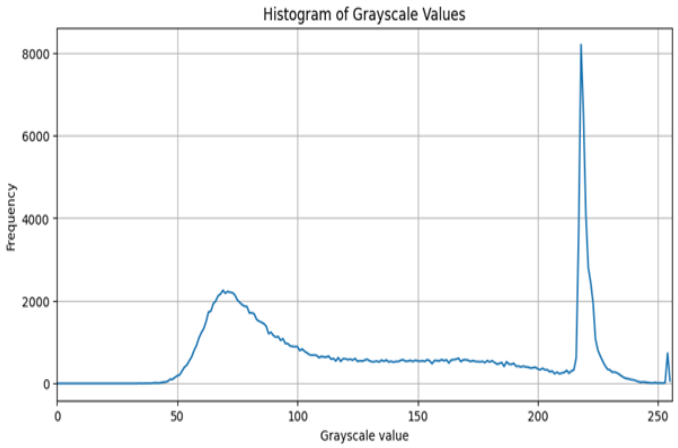


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

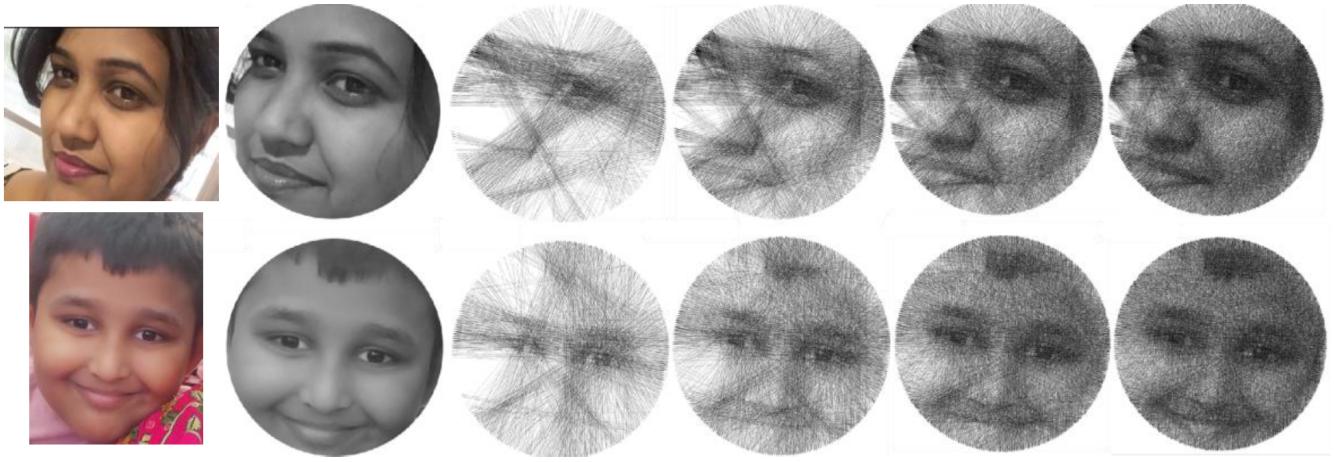


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

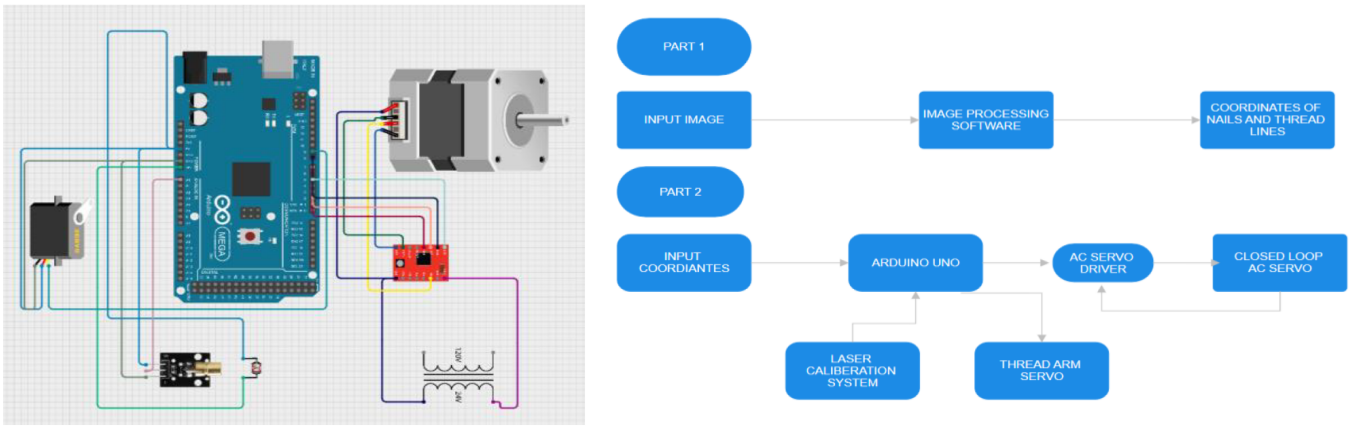


FIGURE 7: Circuit connection with block diagram representation.

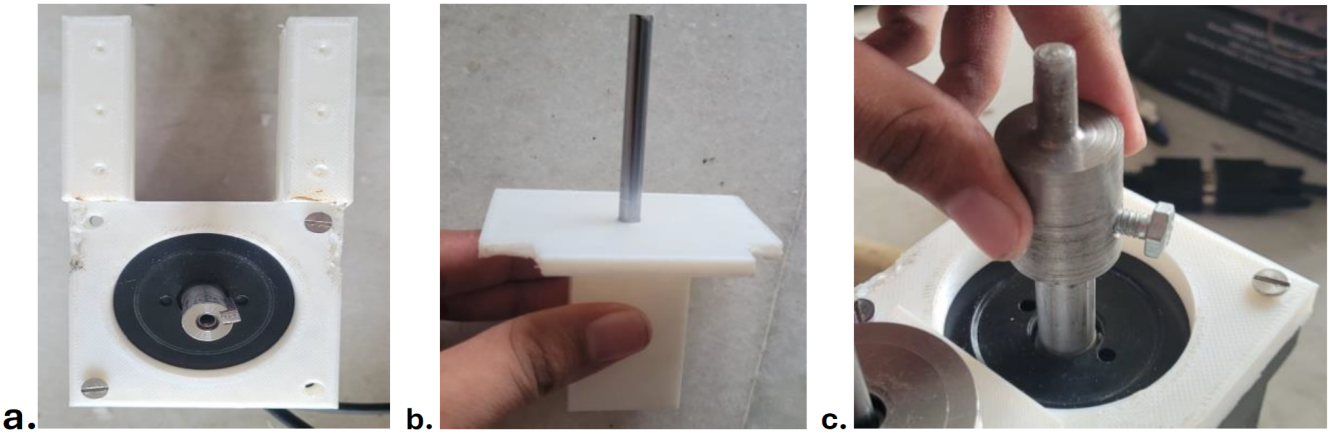


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

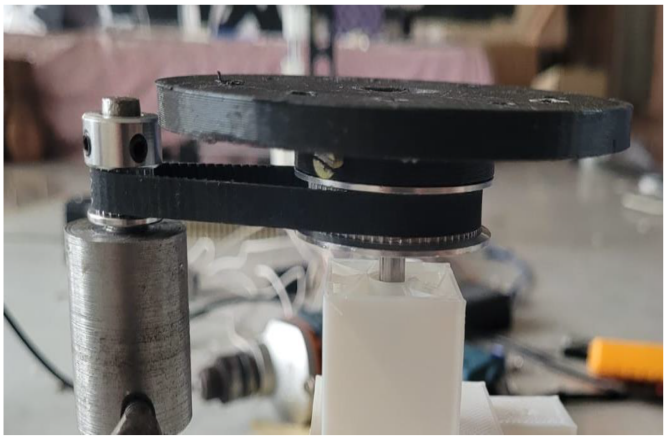


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

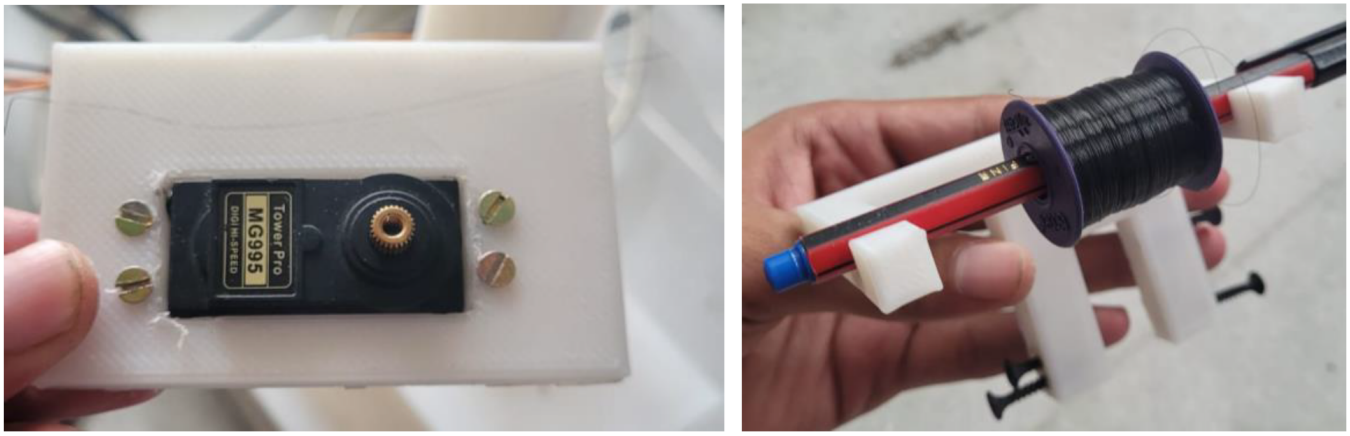
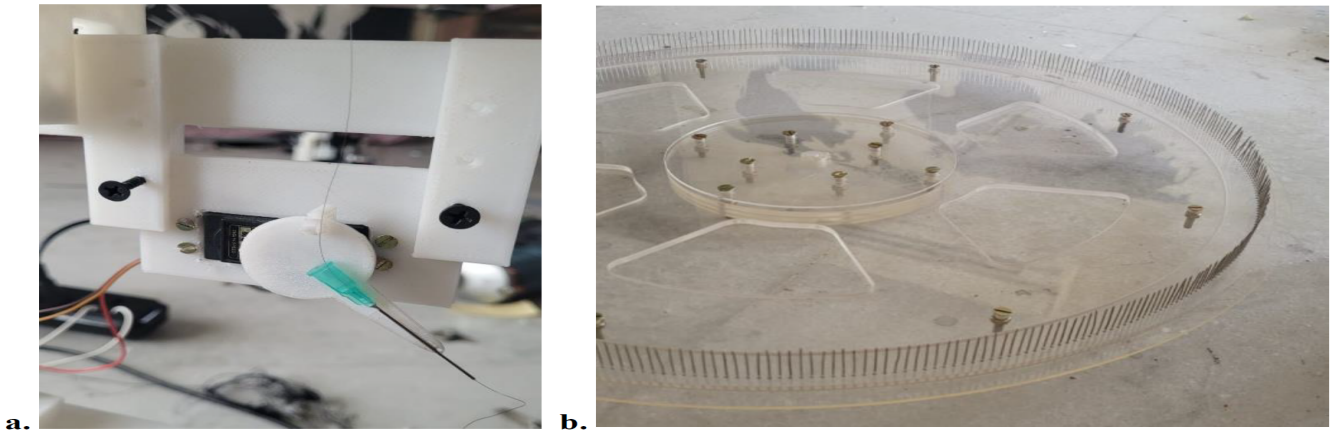


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

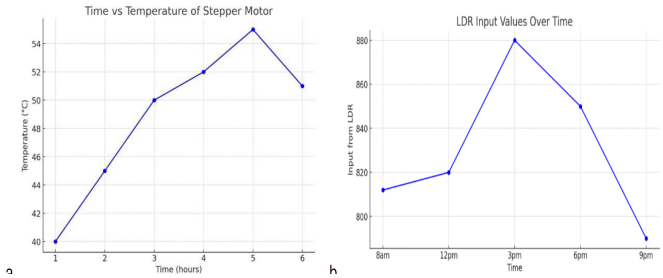


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

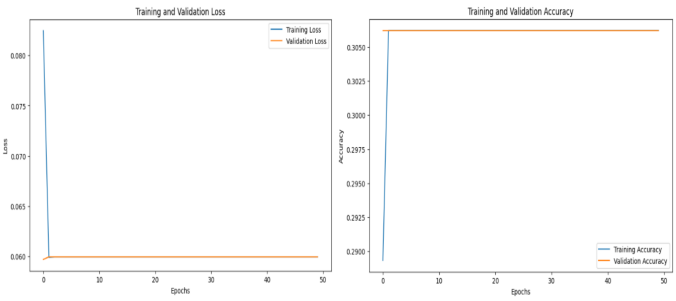


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Automated String Art Creation: Integrated Advanced Computational Techniques and Precision Art Designing

SPOORTHI SINGH[1], NAVYA T HEGDE[2], MOHAMMAD ZUBER[2], YUVRAJ SINGH NAIN[1], and VISHNU G NAIR[2*]

[1]Department of Mechatronics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104
[2]Department of Aeronautical and Automobile Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, Karnataka, India-576104

Corresponding authors: Vishnu G Nair (e-mail: vishnu.nair@manipal.edu)

ABSTRACT The Thread Art Machine project automates the traditional, labour-intensive process of string art creation by integrating advanced computational and manufacturing techniques. Utilizing CAD models in Fusion 360, CNC machining, 3D printing, and Arduino programming, the machine precisely sets threads and nails to form intricate string art patterns. Key components of the machine include adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for image processing, which enhance both the accuracy and aesthetic quality of the final artwork. The project faced challenges such as material selection, CAD design, and hardware-software interfacing, all of which were addressed through iterative design and validation processes. A convolutional neural network (CNN) was employed to process grayscale images, extracting and reconstructing features using pooling and deconvolution techniques, with the model achieving stable performance over multiple epochs. The machine's calibration system, involving LDR sensors and laser alignment, ensures precision in thread placement. The resulting string art pieces, derived from famous images, demonstrate the machine's capability to capture key features through varying string densities, offering a blend of mathematical precision and artistic abstraction. Validation through training and testing revealed consistent performance, with minimal overfitting, as indicated by flat training and validation loss and accuracy curves.

INDEX TERMS String Art Automation, String Art and Technology, Digital Fabrication, Precision Manufacturing, Computational Art, Algorithmic Design, Robotics in Art.

I. INTRODUCTION

RECENT advancements in technology have facilitated the fusion of new-age tech with traditional artisan techniques, giving rise to innovative forms of creative expression. One such example is the String Art Generator, which combines digital image processing with the conventional craft of string art to produce aesthetically pleasing artworks. Traditionally, string art involves placing nails or pins on a board and threading them to form designs. This project aims to automate this process, thereby making it more efficient and opening new possibilities for artistic creation. The integration of image processing techniques with string art has been a focus for researchers, who aim to simplify the process of converting digital images into string art patterns. This approach not only saves time but also allows for the creation of more complex and diverse artistic works. The String Art Generator project stands out by enabling users to upload their own images and customize various parameters, thus democratizing the art form and encouraging public engagement. As a motivation, The Thread Art Machine project seeks to innovate the process of automating string art through modern technology. Key areas of focus include the mechanical design and fabrication of system components using tools like Fusion 360, 3D printing, and CNC milling. The project also emphasizes the importance of electronics and programming in achieving precise control over the machine's operations. Additionally, the development of new algorithms for pattern generation and optimization is crucial for this project. These algorithms must efficiently convert digital images into directives for the machine, ensuring accurate and aesthetically pleasing results.

The project aims to demonstrate the machine's potential in various applications, including art, special furniture, and educational tools.

The traditional process of creating string art is complex, time-consuming, and requires significant manual effort. This limits the ability of artists to produce intricate designs and discourages experimentation. The Thread Art Machine project aims to address these challenges by automating the nail placement and string threading processes, thus enabling faster and more consistent creation of string art. In the work" A computational approach for spider web-inspired fabrication of string art" by Seungwoo Je, the authors focus on the deployment of a sting art machine [1]. seeks to detail an exciting new method of automatically creating the string art using the inspiration from the webs developed by the Miagrammopes spiders. This review discusses related research on computational fabrication, making string art, and synthesizing images with respect to the authors' work. the author Igor Ostanin focus on the deployment of a optimal composite Materials and structures [2]. it describes a new solution to ply composite structures' design and manufacturing that enable a structure with maximum stiffness for certain weigh limitation and loading pattern. Incorporating concepts from string art, a method of art that utilizes a separate tensioned string with pins as the image element, the authors use a discrete optimisation algorithm to represent a 2D distribution of isotropic material. These designs are formulated by the integration of string art outlook that is new into the conventional topology optimization methodology to develop optimal composites. This process involves two stages: the distribution was approximated with a polyline, standing for stiff reinforcement fibres in a soft matrix of the material. This fact is explained by numerous examples described in the paper and further considerations of applying this approach to the development of the new generation of fibrous composite structures in the industrial field.

The author [3] focuses to elucidate the mathematical principles underlying string art, particularly in the creation of cardioids and nephroids, through the exploration of modular arithmetic, prime numbers, and primitive roots. String art is very enlightened in the literature, which discusses the application of mathematical concepts and aesthetic design including the use of modular arithmetic and properties of prime numbers to create appealing geometric figures like cardioids and nephroids. the author defines "60-second images" as a class of geometrical patterns that can be produced with the help of Electronic String Art model. The model – based on the motion of a clock's second hand – generates complex pictures made up of 60 similar segments – each including one additional vertex in a regular polygon. Players can adjust parameters of the picture adjusting its structure as well as its appearance: n, S, P, and J. The paper explains mathematical foundations of the ESA model, which focused on the use of modular multiplicative inverse that helps move vertices sequentially. The paper, [4] describe a compiler which target is to link high level of design with low level of industrial knitting. Industrial knitting machines provide high-speed creation of elaborate, continuous, and three-dimensional fabrics with high resolution, but most the associated programming interfaces and functionalities entail specific control of relatively simple knitting processes or steps. To tackle this issue, the authors suggest a compiler that will convert assemblies of high-level shape primitives, for example, tubes and sheets, into machine code. These shape primitives allow users to distort knit objects which would otherwise be impossible to do by altering low-level instructions, they provide ease of controls in timing, size, and form. Machine learning algorithms significantly improve the precision and accuracy of string art portraits. The automated process ensures consistent and repeatable results, which is crucial for large-scale production or when creating multiple copies of the same design. Automation through machine learning drastically reduces the time and manual effort required to create string art. What traditionally might take hours or days can now be completed in a fraction of the time. Advanced algorithms allow for easy customization of designs. Users can input any image, and the machine learning model will generate a corresponding string art pattern tailored to that specific image. GANs and other creative AI models introduce new possibilities in string art, allowing artists to explore abstract and non-traditional designs that might be challenging to achieve manually [4], [5], [6], [7], [8], [9]. In addition, the work goes one step further to verify and try the Thread Art Machine in actual projects. This involves highly testing the machine and its capability, going further to identifying some possible applications whereby they may include the artwork, special furniture, and education facilities. There are plans to work closely with industries and other artists, in order to expose the potential of the machine and to check if it can become profitable. By doing so, the researchers endeavour to illustrate how art technology can revolutionize the emerging field of string art and open doors to innovative solutions for weaving and artistic creations.

One major consideration in the proposed work pertains to the connectivity and application of electronics in systems and control. This involves knowledge and utilization of different machines alongside its components including motor drivers, stepper motors, and Arduino microcontrollers. Engineers design complex computational approaches that would allow the machine to guide the accurate positioning of nails and threads as the strings are deployed to produce patterns similar to string art. the development of new and highly complex patterns of image generation and the optimization of walking paths will also be a part of this project. Central to the machine's functionality are adaptive algorithms for real-time thread tension adjustments and transposed convolution layers for advanced image processing, both of which significantly enhance the accuracy and aesthetic quality of the final artwork. Throughout the development process, challenges such as material selection, CAD design, and hardware-software integration were systematically addressed through iterative design and validation. A convolutional neural network (CNN)

was employed to process grayscale images, utilizing pooling and deconvolution techniques to extract and reconstruct key features, with the model showing stable performance over multiple training epochs. Precision in thread placement is maintained through a sophisticated calibration system involving LDR sensors and laser alignment. The resulting string art pieces, inspired by iconic images, showcase the machine's ability to capture essential details through varying string densities, achieving a harmonious blend of mathematical precision and artistic abstraction. Extensive validation, including training and testing, demonstrated consistent performance with minimal overfitting, as evidenced by stable training and validation loss and accuracy curves.

II. METHODOLOGY

To address the challenges of traditional string art and achieve the algorithms that can be utilized, and process of design implementations are discussed as follows. Thread art portraits, also known as string art or thread painting, are intricate artworks created by weaving thread or string around a series of pins or nails on a surface. Algorithms play a crucial role in designing these complex patterns, ensuring precision and visual appeal. Here are some possibilities of algorithms used for thread art portraits: As in fig 1,

- Circle-based Algorithms: Circular Loom Algorithm: This algorithm uses a circle as the base and evenly distributes pins around the circumference. The thread is then stretched between pins based on a predefined sequence or mathematical function to create intricate patterns. Epicycloid and Hypocycloid Curves: These are mathematical curves generated by tracing a point on a circle as it rolls around another circle. They can be used to create interesting and visually appealing patterns.
- Line Drawing Algorithms: Bresenham's Line Algorithm: Commonly used in computer graphics, this algorithm can be adapted to determine the straight-line path of the thread between pins, ensuring minimal gaps and overlaps. DDA (Digital Differential Analyzer) Algorithm: Another line drawing algorithm that can be used to plot points along a straight line between pins.
- Image Processing Algorithms: Edge Detection: Algorithms like the Canny edge detector can be used to identify the prominent edges in a photograph, which can then be translated into paths for the thread to follow. Grayscale Conversion and Thresholding: Converting the image to grayscale and then applying thresholding helps in simplifying the image, making it easier to determine where the thread should be placed. Graph-based Algorithms: Minimum Spanning Tree (MST): Used to connect pins with the minimum total length of thread while ensuring that all pins are connected in some way. Hamiltonian Path: Finds a path that visits each pin exactly once, which can be used to create a continuous thread art piece without breaks.
- Optimization Algorithms: Genetic Algorithms: These can be used to evolve thread patterns over multiple iterations, optimizing for visual appeal or other criteria. Simulated Annealing: A probabilistic technique used for finding an approximate global optimum, which can help in refining the thread path to achieve a more aesthetically pleasing result

Algorithms used in can be adapted to impose the aesthetic of thread art onto a photograph. Implementing these algorithms typically involves several steps, including image preprocessing, determining pin positions, calculating thread paths, and then physically translating these paths into the actual thread art. The choice of algorithm depends on the desired complexity, style, and visual outcome of the final portrait. The initial element is System Design and Architecture, which includes both electrical systems and mechanical design. As a Component Designer, the major task will be to design the mechanical parts of the machine, such as the main wheel, nail placement mechanism, and string threading equipment, using Fusion 360.The utilization of 3D printing and CNC milling for manufacturing components is highly significant. The solution is in utilizing 3D printing techniques for intricate components and CNC machining for accurate features. Choosing the appropriate motor is crucial. For accurate positioning of nails and threading of strings, it is essential to select suitable stepper motors and motor drivers. The incorporation of Arduino or microcontrollers to supervise motor control and sensor inputs, ensuring the synchronized functioning of all mechanical elements.

Regarding the field of Software Development, the algorithm for image processing has been ranked second. The first step in designing the algorithm entails creating a software application that can turn digital images into string art patterns. The technique necessitates the application of edge detection, the refinement of nail placement, and the computation of string path.

Python has been used here for image processing with a web-based interface. This interface should streamline the process of uploading images and enable users to modify certain attributes, such as the number of nails and the colour of the string. Backend Processing, to efficiently manage the uploading of images, perform complex algorithms, and display the outcomes, it is advisable to develop server-side processing using a backend framework such as Django or Flask. When automating the production of string art, it is crucial to consider the strategies used for arranging nails. Creating and implementing a method for precisely positioning nails by leveraging the pattern produced by the image processing algorithm. This requires the creation and execution of the nail implantation mechanism as shown in algorithm 1.

To ensure precise placement of nails at specific locations, it is important to program the stepper motors. String Threading is the act of manipulating and working with strings in a consecutive fashion. As a string path optimization, It is crucial to develop an optimization algorithm that can discover the most efficient way for threading strings, with the goal of minimizing string crossings and maximizing pattern accuracy concurrently. Developing a threading apparatus that

Algorithm 1 String Art Pattern Generation Algorithm

1: **Start**
2: Load Image and Resize Image
3: Apply Gaussian Blur
4: Initialize Nails and Pattern
5: **for** each Thread **do**
6: Set current nail
7: Calculate Line Initialization position
8: **end for**
9: Calculate the time required
10: Plot string art pattern
11: **End**

use stepper motors and is specifically engineered to precisely track the most efficient trajectory of a string. Following are the approaches made for image conversion and feature extraction, while converting it to string art image.

$Y[i,j,k] = \sum_{m=0}^{H-1} \sum_{n=0}^{W-1} X[i+m, j+n, c]$*K[m,n,c,k]+b[k] (1)

Where ,

Y[i,j,k] : is Output feature map at position (i,j) for the kth filter

X[i+m,j+n,c]*[i+m, j+n, c]*[i+m,j+n,c] : is Input image or previous layer's output at position (i+m,j+n) for defined channel c.

K[m,n,c,k]: Kernel weight for the kth filter at position (m,n) for channel c.

b[k] Bias for the kth filter.

H and W: Height and width of the kernel.

The provided above equation (1), it outlines key operations of a convolutional neural network (CNN) model, commonly used in image processing tasks. The Convolution Operation equation describes how each pixel in an output feature map is calculated by applying a convolutional filter (kernel) over the input image, combining the values of the image and the kernel, and adding a bias term. The Convolutional layers have been applied with a set of filters (kernels) to the input image to extract features such as edges, textures, and patterns in python code. where, each kernel slides over the image (or feature map from the previous layer), performing element-wise multiplication with the input values, and then sums them up to produce a single output value. This operation is repeated for each position in the image, generating a feature map that highlights specific characteristics detected by the kernel. Along with that, the bias term is added to each computed output value.

$Y[i,j,k] = \max_{m,n} (X[2i + m,\ 2j + n,\ k])$ .....(2)

Y[i,j,k] : output after max pooling at position (I,j) for the kth feature map.

X[2i+m,2j+n,k] Input value at position (2i+m,2j+n) for the kth feature map.

$\max_{m,n}$: Maximum operation over the window.

The Applied Pooling Operation, specifically Max Pooling, as in equation (2) , reduces the spatial dimensions of the feature maps by selecting the maximum value within a pooling window, thereby retaining important features while reducing computation. The Transposed Convolution or deconvolution operation is used to up sample the feature maps, reversing the effect of pooling or down sampling, which helps in reconstructing the image. The Mean Squared Error (MSE) Loss function is a common metric used to quantify the difference between predicted and actual values, guiding the training process to minimize errors. Finally, the Sigmoid Activation function maps input values to a range between 0 and 1, often used for binary classification tasks or to introduce non-linearity into the model. Together, these operations form the foundation of a CNN's ability to learn and generate complex features from input data [10], [11], [12], [13], [14], [15], [16], [17], [18].

III. SIMULATIONS OF THREAD / STRING ART FOR IMAGE CONVERSION

The first step is the acquisition of the desired input image; it is then converted to grayscale (sigmoid function) and modified to reduce the level of noise in the image using Gaussian blur. Contours are vital in this stage normally using the Canny edge detectors to detect the edges of the objects in the image. This entails gradient computation, which is followed by non-maximum suppression, double thresholding, and finally edge following through hysteresis. After edge detection, features like the critical points, lines and shapes are obtained by applying techniques including the Hough Transform. The image is then divided into several smaller sections that can be easily addressed to achieve the best outcome. The details of this process include the calculation of precise coordinates for nail insertions as well as thread paths in consideration of the edges and features detected to generate an accurate and presentable final image. These coordinates are optimized and given as data to help in the physical making of the string art, thus the image processing software is a crucial part of the Thread Art Machine. The Thread Art Machine project mechanizes the entire process of string art creation, converting the labour-intensive procedure into an efficient, technology-driven undertaking. The method commences by importing an image into the image processing system, which is programmed using the Python programming language. The system utilizes image analysis to determine the essential characteristics required for reproducing the image using string art, such as the exact coordinates for the placement of nails and threads. These coordinates are computed and established as reference points for further activities.

The simulations results obtained out of the program is a histogram plot representing the distribution of grayscale values in the image. The x-axis of the plot represents the grayscale values, ranging from 0 to 255. In a grayscale image, 0 represents black, and 255 represents white, with varying shades of Gray in between. The y-axis represents the frequency of each grayscale value in the image, i.e., how many pixels in the image have that specific grayscale value with sigmoid function. The input image is first converted to a grayscale image. This simplifies the data by reducing it to a

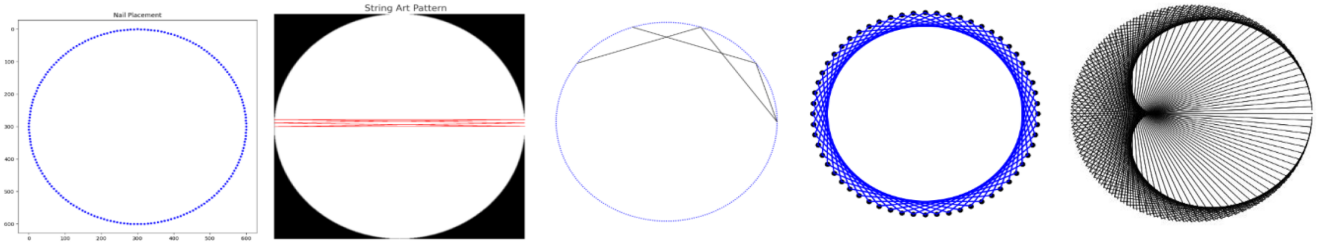


FIGURE 1: Nail or pin placement pattern and Sample String art patterns.

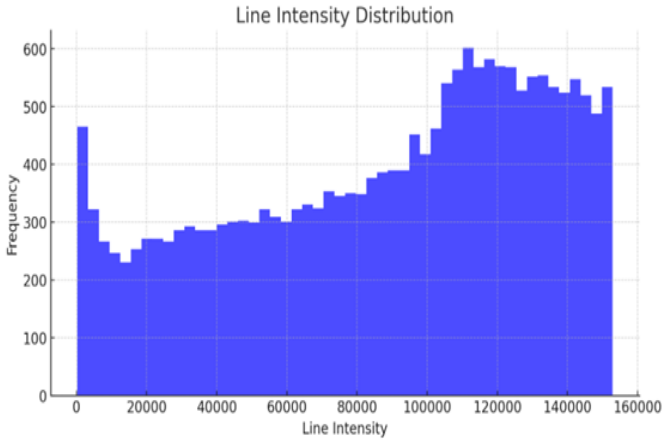


FIGURE 2: Line intensity distribution.

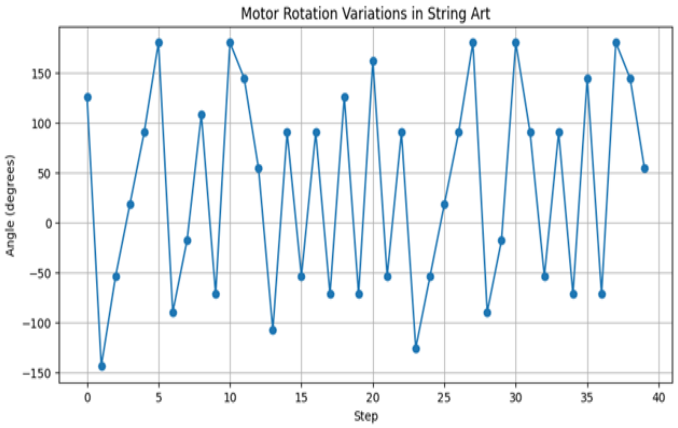


FIGURE 3: Motor rotation variations in string art.

single channel (intensity), as opposed to three channels (red, green, and blue) in a colour image. The grayscale image is then converted into a NumPy array. Each element of this array represents the intensity value of a pixel in the grayscale image. The histogram is calculated by counting the number of occurrences of each grayscale value (from 0 to 255) in the flattened NumPy array. This gives a frequency distribution of pixel intensities.

Figure 1 shows the placement of nails evenly spaced around a circular border. Nails are positioned using polar coordinates, converted to Cartesian coordinates. As a Specifications, 200 nails around a circle with a radius of 300 units. Uniform Distribution**,** Ensures even coverage for the string art pattern. Symmetry**,** Provides a consistent framework for complex designs. Fig 3. Also shows Final visualization of the string art pattern created by connecting nails based on intensity. Lines are drawn based on the highest intensity connections. 200 threads used to form the pattern, Creates intricate and visually appealing designs. Easily adjust the number of nails and threads for different patterns and levels of detail. As in figure 1 the function generates the positions of the pins in a circular pattern using trigonometric functions. The number of pins and the radius of the circle are specified as parameters takes the x and y positions of the pins and a list of connections. Each connection specifies a pair of pins that should be connected by a string. The function plots the pins and the strings on a graph using matplotlib. The main function sets the number of pins and the radius, generates the pin positions, defines the connections, and then plots the string art. Figure 2 shows Line Intensity Distribution**,** showing the distribution of intensities for lines connecting pairs of nails. Intensity values represent the sum of pixel values along a line in the blurred image. Calculated for all unique pairs of nails. Helps identify which connections contribute most to the design. Guides in selecting high-intensity lines for impactful patterns.

The plot at figure 3, shows the variation in angles between consecutive pins as the string moves along the path. Each point on the plot represents the angle between two consecutive pins in the string path. The variation in angles indicates how much the motor would need to rotate at each step to move from one pin to the next. As the angles show sharp changes or large differences between consecutive steps, it indicates that the string is making large or abrupt changes in direction. But Smoother transitions with smaller angle variations suggest a more gradual change in direction, which might indicate a smoother pattern in the string art. The column Layer (type) from Table 1 simulation output lists the layers of the model and their types (e.g., Conv2D, MaxPooling2D, Conv2DTranspose). The column Output Shape shows

the shape of the output produced by each layer. For instance, after the first Conv2D layer, the output shape is (None, 256, 256, 64), which means the layer produces 64 feature maps of size 256x256. None represents the batch size, which is flexible and not fixed during model definition. The column Param #shows the number of trainable parameters (weights and biases) in each layer.

TABLE 1: CNN architecture used in python program for string art generation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 64)	640
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0
conv2d_1 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0
conv2d_2 (Conv2D)	(None, 64, 64, 256)	295,168
conv2d_transpose (Conv2DTranspose)	(None, 128, 128, 256)	295,040
conv2d_transpose_1 (Conv2DTranspose)	(None, 256, 256, 128)	73,792
conv2d_3 (Conv2D)	(None, 256, 256, 1)	577
Total params:	739,073 (2.82 MB)	
Trainable params:	739,073 (2.82 MB)	
Non-trainable params:	0 (0.00 B)	

The model is trained on a very small dataset (just ten image pair), which is duplicated to simulate a larger dataset. As a result, the model's ability to generalize is extremely less. The model learns to map the input image to the output string art image, but with only ten pair of data, the training process is very minimal in terms of learning general features or patterns. The model described in table 1, is a convolutional neural network (CNN) designed to process a grayscale image through multiple layers of convolution and pooling, followed by transposed convolution layers for image reconstruction. It begins with a Conv2D layer with 64 filters of size 3x3, extracting features and outputting 64 feature maps of 256x256. The MaxPooling2D layer then reduces the spatial dimensions to 128x128. This is followed by a Conv2D layer with 128 filters and another pooling layer, further reducing the feature maps to 64x64. A deeper Conv2D layer with 256 filters enhances feature extraction, and two Conv2DTranspose layers sequentially increase the spatial dimensions back to 128x128 and 256x256 with 128 and 64 feature maps, respectively. The final Conv2D layer reduces the channels to 1, producing a single grayscale output image of 256x256. The model has 739,073 trainable parameters and is designed for tasks like reconstructing an image, such as generating a string art image.

The histogram of figure 4, shows the distribution of grayscale values in the image. Each point on the plot represents the number of pixels in the image that have a specific grayscale value. A grayscale value of 0 corresponds to black, 255 corresponds to white, and values in between correspond to various shades of Gray. For Feature Extraction, the utilized Convolutional layers from equation (1), are crucial for learning and detecting important features in images, such as edges, shapes, and textures. Since the same kernel is used across different parts of the image, the number of parameters

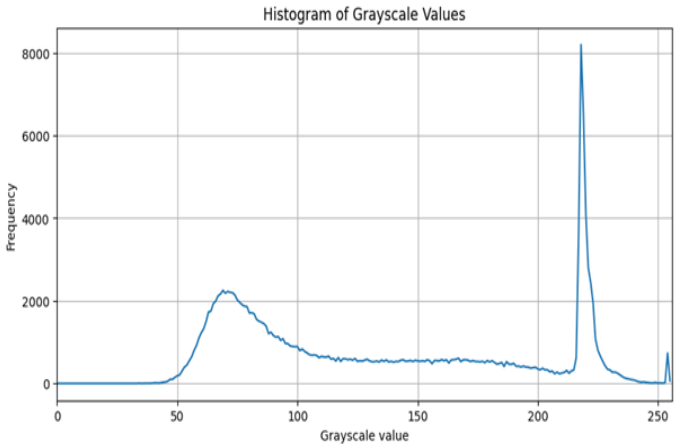


FIGURE 4: Histogram of the grayscale values.

is significantly reduced compared to fully connected layers, making the model more efficient. Max pooling from equation (2) reduces the size of the feature maps, decreasing the computational load in subsequent layers. Translation Invariance regards, Pooling helps make the network more robust to small translations or distortions in the input image.

The process of teaching a machine learning algorithm to convert colored images into string art involves several critical steps. First, a dataset is considered from Bin Yan et al. [19], consisting of paired colored images and their corresponding string art representations from figure 5. The colored images serve as inputs, while the string art images, typically in binary form, act as the target outputs. Preprocessing is essential, where images are resized to uniform dimensions, normalized, and edge-detected to enhance learning. The problem is treated as an image-to-image translation task, and architectures like U-Net or Pix2Pix are employed. These models use convolutional neural networks (CNNs) to learn spatial features from the input and predict string art images as output. Training is performed using loss functions such as pixel-wise loss (e.g., MSE) and potentially perceptual loss to ensure similarity between generated and target images. During training, data augmentation techniques like random flips, rotations, and noise addition help improve model generalization. Once trained, the model is tested on new unseen images, where it predicts string art versions. The performance is evaluated using metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) to measure the quality of the output. This approach leverages deep learning's capability to learn complex mappings and provides a way to automate the artistic process of string art generation from any given colored image [19], [20], [21], [22].

To validate the fidelity of the generated string art images, PSNR and SSIM metrics were calculated for 10 benchmark images. Table 2 shows these results. The results demonstrates high fidelity (PSNR 33 dB) and structural similarity (SSIM 0.996) between the generated images and their input references.

FIGURE 5: Data set used for teaching the machine learning algorithm [19].

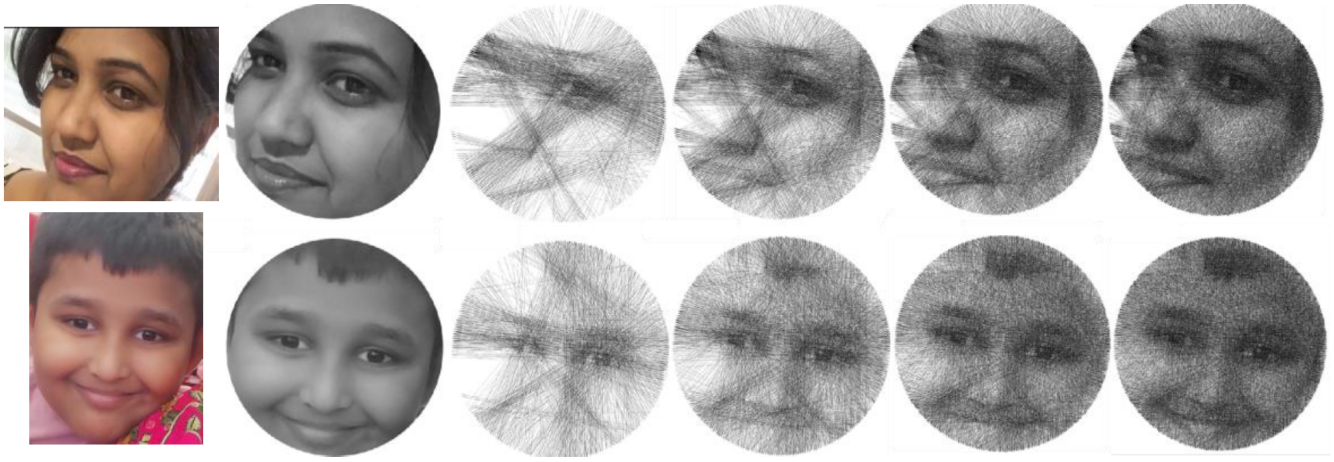


FIGURE 6: Coloured image to greyscale conversion and string development art at different completion stages (simulation results).

TABLE 2: Quantitative Metrics for Reconstructed String Art Images: PSNR and SSIM metrics

PSNR (dB)	SSIM
32.97259	0.996964
32.95046	0.996938
32.94167	0.996931
32.97444	0.996959
32.96877	0.996942
32.97440	0.996948
32.97080	0.996953
32.96390	0.996962
32.97270	0.996948
32.95573	0.996949

After teaching set of data for string art conversation, the sample colored image used to obtain the simulated string art image is as shown in figure 6. The sample string art simulated image for the set value as, number pins = 288, number of lines = 2500, line weight 18 , as per figure 6 the obtained coloured to greyscale converted image with circular cut is then string art converted. They are with 709 lines during 28% of its image building, 1277 lines during its 51% of image building, 1906 lines during its 76% of image building and 2500 line utilized for 100% of its image building in its string art form are as shown in figure 5.

The hyperparameter settings for our convolutional neural network (CNN) were meticulously chosen to balance per-

formance and computational efficiency. A learning rate of 0.001 was set, which ensures a steady and stable convergence during training, minimizing the risk of overshooting the optimal solution. The Adam optimizer was employed due to its adaptive learning capability, incorporating momentum terms (1=0.9,2=0.999) that accelerate convergence and improve robustness to noisy gradients. The mean squared error (MSE) loss function was selected to penalize large deviations between the predicted and actual images, guiding the model to accurately reconstruct fine details in the string art. A batch size of 16 was used, striking a balance between computational efficiency and gradient stability, while training the model for 100 epochs allowed sufficient iterations to learn complex patterns without overfitting. The CNN architecture consisted of convolutional filters of sizes 64, 128, and 256 across layers, which progressively extracted features of increasing complexity. A 3x3 kernel size was chosen to effectively capture spatial patterns, such as edges and textures, without excessive computational cost. MaxPooling with a 2x2 window was used to reduce spatial dimensions while retaining salient features, optimizing both memory and processing requirements. The ReLU activation function was applied to introduce non-linearity, essential for learning complex mappings, while a Sigmoid activation function in the final layer ensured the output values were normalized between 0 and 1, suitable for binary classification tasks. All model development was implemented in TensorFlow/Keras, leveraging its high-level abstraction for rapid experimentation and robust implementation. These carefully tuned hyperparameters collectively contributed to the model's effectiveness in generating accurate and visually appealing string art representations.

IV. HARDWARE DESIGN SETUP

The physical construction of the Thread Art Machine is a piece of equipment that represents a well-arranged assembly of numerous elements and parts as shown in figure 7. It is created through the most innovative of technologies, with particular emphasis on the proper combination of numerous effective techniques that would allow for precise and successful creation of the Thread Art Machine. Some subassemblies are represented by 3D elements; they ensure individual and confined forms to correspond with the peculiarities of the machine. These 3D printed sections make it lightweight yet strong and can be easily adjusted and adapted if necessary. The timing pulley and belt system is also included in the machine setup, which plays an essential role in regulating the accurate positioning of the thread assembly via a tensioned belt and pulley combination. Laser cut parts are used where there is accuracy and smooth surfaces needed in the components as well as increasing the solidity and visual appeal of the machine. The incorporation of all these aspects brings a qualitative integration of mechanical and electronic components thus making the Thread Art Machine an efficient tool in the creation of string art patterns.

A real-time system state estimator consistently updates the precise position and orientation of the machine by integrating newly acquired sensor data. This guarantees that the robots maintain a precise position and are correctly monitored as they move across the designated area. The coordinates obtained from the image processing tool are subsequently inputted into an Arduino Uno microcontroller. The Arduino Uno receives the coordinates and controls the arrangement of the thread and the functioning of the stepper motor for the string art project. A limit switch is used to send feedback to the Arduino Uno, controlling the movement of the thread assembly. This feedback mechanism guarantees the accurate positioning and tightening of the string, leading to the creation of top-notch string art designs that are derived from the input image. The Thread Art Machine project showcases the integration of art and technology by utilizing advanced technologies including Python programming, real-time state estimate, and precise hardware control with Arduino. This method not only automates the production of string art, but also improves the accuracy and effectiveness of the process, delivering impressive outcomes that match the original image requirements.

The MOTOR HUB part as shown in figure 8, is designed and modelled through fusion 360. It sits directly on the motor. It is screwed to the motor with the help of four M4 screws. The extended arm of this hub hosts the slider mechanism for the wheel shaft hub to slide. The sliding mechanism of this part helps to adjust the tension in the timing belt. It took me two tries to build the exact needed part. This part is 3D printed with PLA which is light and sturdy. Shaft holder is also designed and 3d printed in the software fusion 360. This part holds the main wheel's shaft. On which the timing gear of the wheel is mounted. This part directly slides into the motor hub. The sliding mechanism ensures that the tension in the timing belt remains constant. Regarding Motor shaft mount: The diameter of the shaft of the motor designed is 14mm and the inner diameter of the driving timing pulley size is 8 mm. To mount the timing pulley on the motor, a metallic hub is utilized on the lathe machine. The prototype designed on fusion 360 and the original metal piece made 3D printer.

The Timming pulley and Belt , as in figure 9, is used for power transmission, which allows precise moment of the driven shaft. The gear ratio between the driving pulley and driven pulley is 1:3 this increases the torque of the driven pulley. The driving pulley which has 20 teeth is mounted on the motor and the driven pulley which has 60 teeth's is mounted on the shaft holder. The driven pulley has rolling bearings LM8UU fitted inside it for smooth and precise motion around the shaft. Servo holder of the Thread assembly as in fig 12, is also designed, and 3D printed. This is also made of PLA material. This part has a sliding mechanism. The servo motors plate slides into this part. The sliding mechanism helps us to adjust the height of the servo motor. Servo slider block is the hub of the servo motor of the thread assembly. This part is designed in Fusion 360 and 3D printed. This part directly slides into the servo holder. The

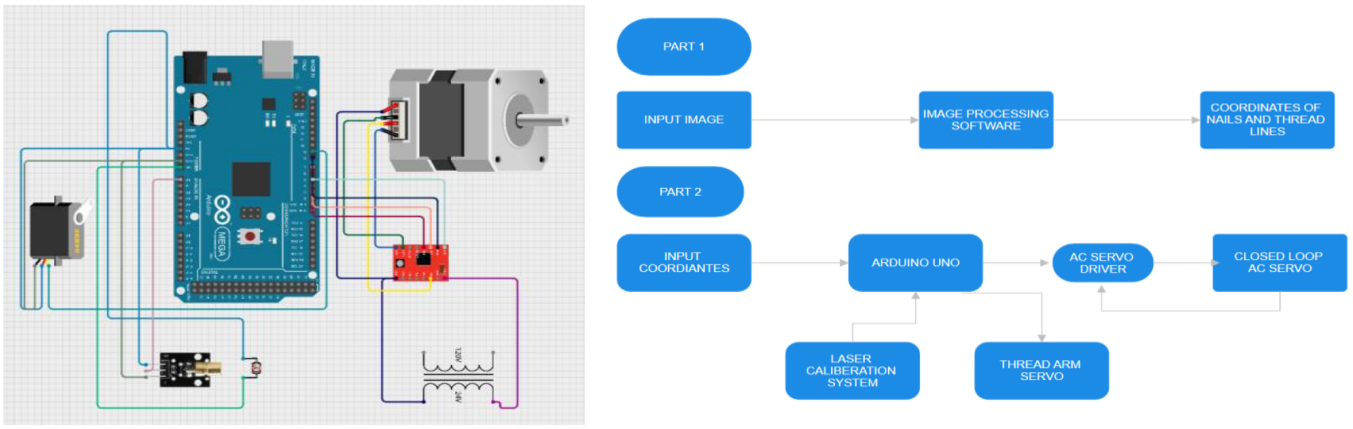


FIGURE 7: Circuit connection with block diagram representation.

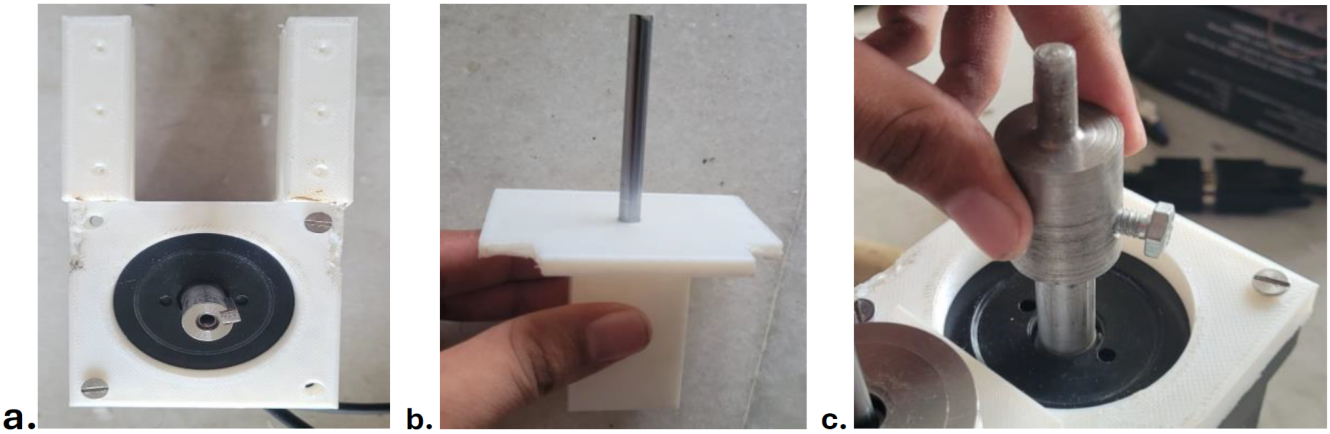


FIGURE 8: Motor hub, shaft holder & motor shaft mount.

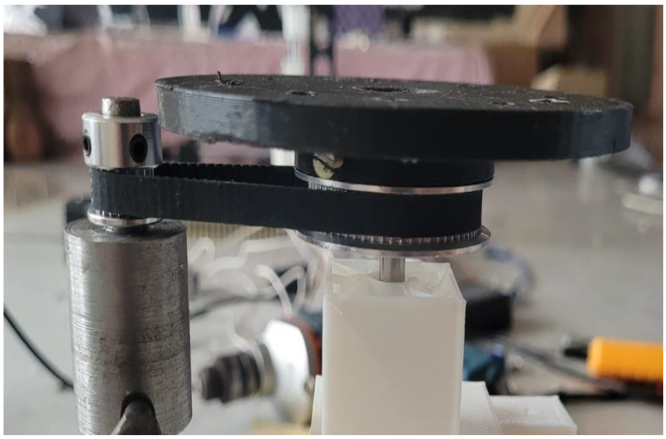


FIGURE 9: Timming pulley and belt.

sliding mechanism helps in adjusting the height of the thread assembly.

Thread hub (10) is mounted on the servo holder. This mount has 4 screws and it mounts on the top. The screw mounting mechanism helps in adjusting and positioning of the thread mount. This mount holds the shaft in which the thread roll is inserted. Servo mount as in figure 11, is mounted on the servo. It has a needle of the syringe attached to it. The thread coming from the thread roll passes through this syringe. Laser cut acrylic wheel, is the main wheel which is driven. The workpiece is kept on top of this wheel, which got designed in fusion 360 and cut from the laser.

AC servo motor is a precision motor type, which is a loop motor driven with an AC current. It takes input from a motor driver which directs the motor in a desired manner according to the feedback signals. This makes the control of position, velocity as well as torque possible since the system forms a closed loop one hence cut on accurateness. Speed and position signals are given to the AC servo motor by the motor driver. This, however, has a downside in most motors that come with sensors, commonly encoders that feedback the motor status to the driver in real-time. This feedback loop enables the motor driver to act almost instantaneously to correct any variations from the required track. The detailed fully working of physically assembled servo mount with thread pin

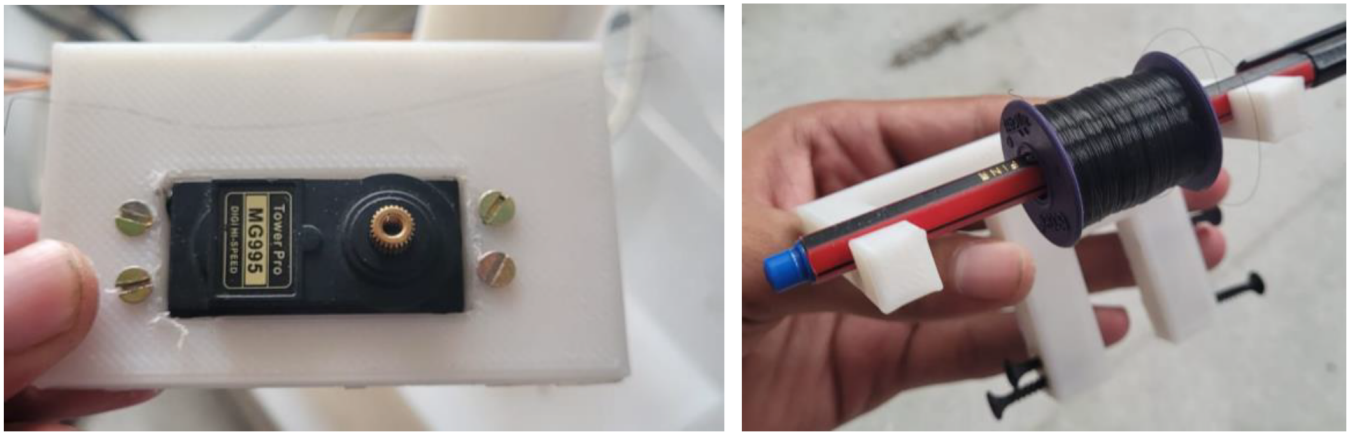
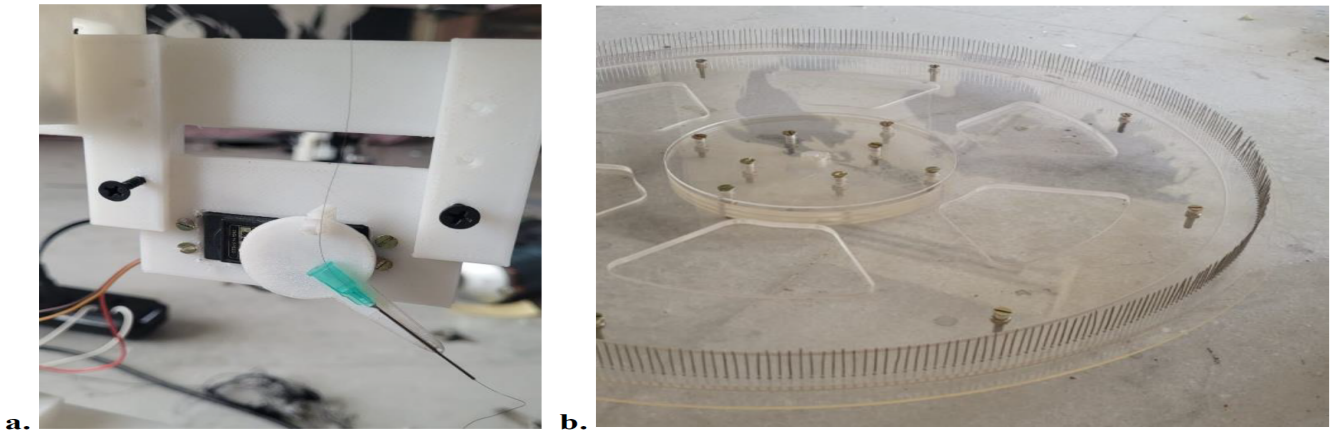


FIGURE 10: Servo slider block & Thread hub.



a. b.

FIGURE 11: Servo mount & acrylic wheel.

for given image can be seen in figure 12 below. Originally, the stepper motor is mechanically very inefficient, and this causes the dissipation of a lot of heat. Stepper motor being very inefficient due to its precise degree of rotation hand high holding torque it dissipates lots of heat. the readings of temperature were taken at different time interval, and it is plotted in figure 13a.

Laser is installed perpendicular to the Light Dependent Resistor (LDR) sensor. The above two types of the diodes are employed for calibration of the machine / system after every 20 lines have been processed. When the laser falls right on the LDR its reading is some higher amount of value i. e. 800 – 900. When the given material is covered with something the LDR provides a low value through which Arduino is informed on the position of nail one. This calibration system is integrated in a close loop with the Arduino system. There is a difference in the higher values of LDR at some time of the day because the amount of daylight affects the LDR. The idle values of LDR sensor varied during different time period of the day. This was because of the daylight in the room. Some observations were taken thought the day and are tabulated and plotted as shown in figure 13b.

The most crucial reason for plotting above curves is to detect overfitting. Overfitting occurs when the model performs well on the training data but poorly on unseen (validation) data. By comparing the training and validation loss, it is convenient to monitor the model's performance and make informed decisions about adjustments to the model architecture, learning rate, regularization techniques, or early stopping. In above figure 14, the training loss starts relatively high at the beginning (around 0.09) and immediately drops to a value similar to the validation loss after the first epoch. After the first epoch, both the training and validation loss curves become almost flat and remain constant throughout the remaining epochs. The training loss quickly drops to around 0.06, where it stays for the rest of the training process. The validation loss starts and remains at around 0.06 without much change. From 14, the training accuracy starts slightly below 0.29 and immediately jumps to around 0.305 in the first epoch. After the first epoch, both the training and validation accuracy curves become completely flat and remain constant at approximately 0.305 throughout the remaining

FIGURE 12: Fully working servo mount with thread.

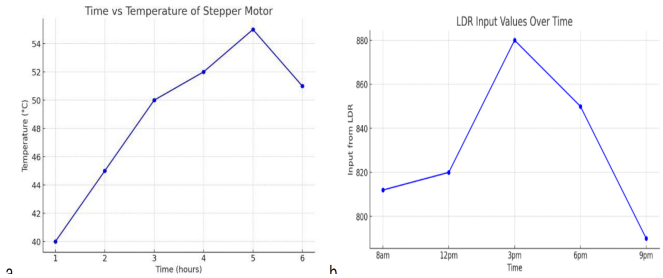


FIGURE 13: Plots of a. motor temperature vs time b. Plot of time vs LDR readings.

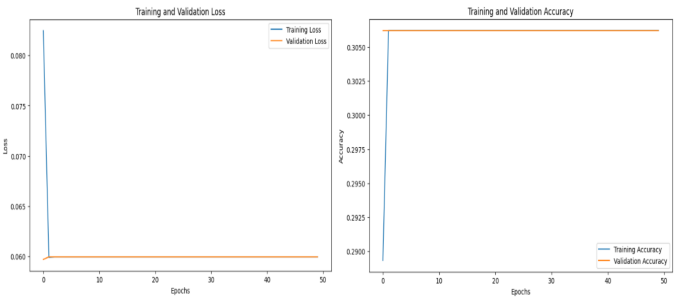


FIGURE 14: Training and validation loss / validation accuracy

epochs.

The top left, middle left, and bottom left corners of the image as in 15, it displays the original images that were used as inputs for the string art process. These include famous images, such as Einstein's face, a portrait of a person, and a colourful lion image. The rest of the images (in each set of three) represent the step-by-step process and results of converting these input images into physical string art. This is typically done using an algorithm that converts the grayscale values of the image into instructions for placing threads on a circular frame with pins. The final string art images as in fig 17, successfully capture the key features of the original photographs. The use of string density to represent varying levels of grayscale allows for recognizable portraits, even in a medium as abstract as string art. The level of detail varies among the different string art pieces. The Einstein portrait and the woman's face, being more straightforward, appear more detailed and recognizable. The lion's face, due to its complex and colourful nature, is more abstract but still identifiable. The transition from fewer strings to more strings adds complexity and detail to the image, gradually forming a clear portrait as more threads are added. String art inherently introduces a level of abstraction, as it cannot capture the full range of colours or details of the original image. Instead, it relies on the viewer's ability to recognize patterns and forms from the density and arrangement of strings. The resulting artworks are unique and have an artistic quality that blends mathematical precision with creative interpretation.

To overcome limitations from a small dataset, we utilized a larger dataset of over 1100 high-resolution images (source: Kaggle). This dataset includes diverse patterns and styles, enabling our model to generalize effectively. Layer-wise visualizations, such as heatmaps, are incorporated to demonstrate feature extraction and learning by the CNN. These visualizations highlight how our model processes critical elements like edges and gradients. In discussing the results of the model's performance, it is essential to emphasize how each of the mathematical operations contributes to the outcomes. The Convolution Operation successfully extracts detailed spatial features from the input image, allowing the model to capture important patterns such as edges and shapes, which are crucial for tasks like object detection or segmentation. The stacked convolutional layers progressively refine these features, making the model adept at identifying more complex structures as it moves deeper through the network. The Pooling Operation helps reduce the size of the feature maps while preserving the most important information, leading to more efficient computation without significant loss in feature quality. This operation contributes to a

FIGURE 15: Real time obtained string art portrait through designed automated machine

lower memory footprint and faster processing, allowing the model to scale for large datasets or high-resolution images. Similarly, the Transposed Convolution layers effectively reconstruct the image's original size during the decoding process, reversing the pooling's down sampling effects. This is particularly important for image generation tasks, such as creating higher-resolution outputs from low-resolution inputs or reconstructing images in applications like super-resolution or segmentation. In terms of optimization, the MSE Loss function ensures the model is accurately learning from its mistakes by penalizing large deviations between predicted and actual values. This drives the model toward producing more precise output images with minimized reconstruction errors. Finally, the Sigmoid Activation function's non-linear behaviour aids in distinguishing features, especially in binary classification tasks or segmentation where a clear decision boundary is needed. It helps the model better approximate complex relationships, leading to more reliable and accurate results. Together, these operations combine to create a robust framework for feature extraction, classification, and image generation, making the model highly applicable to real-world challenges such as medical imaging, object recognition, and image enhancement.

To further emphasize the originality of the proposed work, table 3 presents a comparative analysis of the Thread Art Machine with prior works in the domain. The proosed work is compared with two available ones ([1], [2]), and the advancements in algorithmic sophistication, image processing, and physical implementation are compared. This comparison underscores our contributions in integrating advanced computational techniques with precise hardware control.

TABLE 3: Comparison of proposed Thread Art Machine with [1] and [2]

Feature	**Proposed Thread Art Machine**	**Seungwoo Je (2019) [1]**	**Igor Ostanin (2020)[2]**
Image Processing	CNN with advanced feature extraction	Basic edge detection	Discrete optimization for composites
Automation	Fully automated threading and nail placement	Semi-automated	Manual input required
Hardware Integration	Fusion 360-based CAD, CNC machining, Arduino control	Limited mechanical integration	Not hardware-focused
Novel Algorithms	Adaptive tension adjustment and deconvolution techniques	Epicycloid-based patterns	Optimization algorithms for materials
Output Customization	Dynamic grayscale conversion and string density control	Fixed patterns	Not applicable
Applications	Art, furniture, educational tools	Limited to aesthetic designs	Industrial composite designs

V. CONCLUSION

Throughout the development of the Thread Art Machine, we have employed a wide array of tools and methodologies. CAD models were meticulously crafted in Fusion 360, with parts being fabricated using CNC and 3D printers, ensuring the structural integrity and precision of the machine components. These mechanical components are seamlessly integrated with Arduino-controlled electronic systems, where

Python programming plays a crucial role in automating the threading process. Our approach also incorporates modern string art techniques, which involve parametric design and adaptive algorithms that adjust thread tension and placement in real-time, ensuring that each string art piece is both mathematically precise and aesthetically pleasing. The automation process is further enhanced by laser cutting and advanced 3D printing technologies, which produce uniform and precise frames and pegs, pushing the boundaries of what is possible in digital fabrication and computational art. The challenges faced during this project, including the selection of suitable 3D printing materials, CAD design complexities, hardware-software interfacing, and the development of robust actuation systems, have been met with innovative solutions. These include the development of adaptive algorithms that respond to real-time changes, enhancing the overall accuracy and quality of the final artwork. As demonstrated through the execution of various string art patterns—ranging from simple portraits to complex designs—the Thread Art Machine not only automates the creation of string art but also elevates it, offering new creative possibilities for artists and enthusiasts. By merging art and technology, this project exemplifies the future of digital fabrication, enabling the creation of intricate and detailed string art with unprecedented efficiency and precision.

The Thread Art Machine successfully automates and enhances the string art process, combining art and technology to create detailed and abstract representations of images. The machine's ability to precisely control thread placement and tension through advanced algorithms marks a significant advancement in digital fabrication. Future work will focus on expanding the machine's capabilities, including the use of colour threads for more complex designs, further optimization of the CNN model for better image reconstruction, and integration of more sophisticated sensors to improve calibration and accuracy. These enhancements will push the boundaries of computational art, offering new creative possibilities for artists and engineers alike.

ACKNOWLEDGMENT

The authors thank the Manipal Academy of Higher Education, Manipal, for providing the facilities needed for the proposed research work.

AUTHOR CONTRIBUTIONS

Spoorthi Singh Idea Generation, Software Simulation, Writing. **Navya T Hegde**Software Simulation, editing. **Mohammad Zuber**Idea Generation, Software , Writing, review and supervision. **Vishnu G Nair:** Writing, Formatting, editing, review and submission. **Yuvraj Singh Nain** Experiments and initial writing.

DECLARATIONS

FUNDING AND CONFLICTS OF INTERESTS/COMPETING INTERESTS.

The authors declare that no competing financial interest or personal relationship could have appeared to influence the work reported in this paper. The authors have no relevant financial or non-financial interests to disclose.

On Figures: The faces in figure 5 is taken from the reference paper [19]. The faces used in figure 6 is that of the first author (Spoorthi Singh) and her son.

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Je, Y. Abileva, A. Bianchi, and J. C. Bazin, "A computational approach for spider web-inspired fabrication of string art," Computer Animation and Virtual Worlds, 2019, doi: 10.1002/cav.1904.

[2] I. Ostanin, "“String art” approach to the design and manufacturing of optimal composite materials and structures," Compos. Struct., vol. 246, 2020, doi: 10.1016/j.compstruct.2020.112396.

[3] A. K. R. Abadio et al., "A pharmacophore-based virtual screening approach for the discovery of Trypanosoma cruzi GAPDH inhibitors (vol 5, pg 2019, 2013)," J. Chem. Inf. Model., 2015, doi: 10.1002/9783527633326.

[4] J. McCann et al., "A compiler for 3D machine knitting," in ACM Trans. Graphics, 2016, doi: 10.1145/2897824.2925940.

[5] H. Mao, M. Cheung, and J. She, "DeepArt: Learning joint representations of visual arts," in MM 2017 - Proc. 2017 ACM Multimedia Conf., 2017, doi: 10.1145/3123266.3123405.

[6] Y. Wang, C. Yang, S. Farrell, Y. Zhang, T. Kurth, and S. Williams, "Time-Based Roofline for Deep Learning Performance Analysis," in Proc. DLS 2020: Deep Learning on Supercomputers, Held in conjunction with SC 2020: The Int. Conf. High Perform. Comput., Networking, Storage and Analysis, 2020, doi: 10.1109/DLS51937.2020.00007.

[7] C. Li, K. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," in Advances in Neural Inf. Process. Syst., 2017.

[8] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks," AASRI Procedia, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[10] V. Dumoulin, F. Visin, and G. E. P. Box, "Object detection _ssd : A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [cs, stat], 2018.

[11] M. Birsak, F. Rist, P. Wonka, and P. Musialski, "String art: towards computational fabrication of string images," Comput. Graphics Forum, vol. 37, no. 2, 2018, doi: 10.1111/cgf.13359.

[12] C. von Renesse and V. Ecke, "Discovering the art of mathematics: Using string art to investigate calculus," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124160.

[13] S. Bagai, "Scrutinizing String Art Through a Mathematical Lens," PRIMUS, vol. 33, no. 3, 2023, doi: 10.1080/10511970.2022.2068094.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, doi: 10.1007/978-3-319-10590-1_53.

[15] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electron. Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[16] M. Yousef and J. Allmer, "Deep learning in bioinformatics," Turk. J. Biol., vol. 47, no. 6, 2023, doi: 10.55730/1300-0152.2671.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[18] Y. S. Yan, H. L. Cai, and B. Yan, "Data hiding in symmetric circular string art," Symmetry (Basel), vol. 12, no. 8, 2020, doi: 10.3390/SYM12081227.
[19] L. Koss, "Visual arts, design, and differential equations," J. Math. Arts, vol. 11, no. 3, 2017, doi: 10.1080/17513472.2017.1373326.
[20] M. Jovanović, M. Vučić, B. Tepavčević, M. Raković, and J. Tasevski, "Robotic Knitting in String Art as a Tool for Creative Design Processes," in Advances Intell. Syst. Comput., 2020, doi: 10.1007/978-3-030-19648-6_21.
[21] E. S. Araújo, R. de B. Mota, and C. Y. B. Oliveira, "Teaching Math through Art: a didactic sequence proposal with scripts for the construction of conics by the string art technique," Ensino da Matemática em Debate, vol. 10, no. 2, 2023, doi: 10.23925/2358-4122.2023v10i258993.
[22] R. La Haye, "String art in a first calculus course," PRIMUS, vol. 26, no. 4, 2016, doi: 10.1080/10511970.2015.1124302.

SPOORTHI SINGH (Member, IEEE) received her bachelor's degree in Instrumentation and Control Engineering from Bapuji Institute of Technology [BIET]- VTU-Davangere, Karnataka, India, in 2009. Completed M.Tech from Astronomy and Space Engg. From Manipal Institute of Technology- Manipal,Karnataka,India, in 2012. Recently completed her Ph.D. degree in Aerospace Engineering from Universiti Putra Malaysia-, Serdang-43400, Malaysia in 2024. She has published research articles in international journals (with high impact factors) and at conferences too. Her research interests include flapping wing drone mechanisms and Robotics: design, control, and fabrications. Working as Senior Assistant Professor from 2013 to present date in Mechatronics Department of Manipal institute of Technology [MAHE]-Manipal.

NAVYA T HEGDE received her Ph.D. and MTech from Manipal Institute of Technology Karnataka, India. She is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. Her research interests include control, motion planning and multi-robotic systems.

MOHAMMAD ZUBER received his Ph.D. UPM Malaysia. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning, CFD etc

YUVRAJ SINGH NAIN is a undergraduate student in the Department of Mechatronics Manipal institute of Technology,Manipal, India. His research interests include robotics, design etc

VISHNU G NAIR received his Ph.D. (2019) from National Institute of Technology Karnataka, India, M-Tech (2012) from Manipal Institute of Technology, India and B-Tech (2010) from Kerala University, India. He is currently with the Department of Aeronautical Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. His research interests include control, motion planning and multi-robotic systems.