Arab Open University- Egypt



Faculty of Computer Studies

Information Technology and Computing Department

***< CNC (Computer Numerical Control) Plotting Machine Software >***

Name: Ahmed El Sayed Mohamed El Sayed

ID: 21510822

TM471: Final Year Project, December 2024   
Supervisor:

Dr/Nisreen Osman

Table of Contents:

[Abstract: 5](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744011)

Acknowledgments…………………...……………………………………………….…...…...…5

Chapter one……………………………………………………………………………………….6

1. [Introduction: 7](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744013)

[1.1 Background:………………………………………………………………………………7](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744014)

[1.2 Aims and objectives:………………………………………………………………………7](file:///C:\\Users\\Hp\\Desktop\\TM471-1\\Wadaq%20Tarek%20GP.docx" \l "_Toc120744015)

[1.2.1 Aims…………………………………………………………………………………7](file:///C:\\Users\\Hp\\Desktop\\TM471-1\\Wadaq%20Tarek%20GP.docx" \l "_Toc120744015)

1.2.2 Objectives…………………………………….……………………..……………...8

[1.3 Deliverables:…………………………………………………...……………………….](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744016) 8

[1.4 Problem definition: 8](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744017)

[1.5 Project Scope: 9](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744018)

[1.6 Target Customer: 9](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744019)

1.7 Suggested solutions:……………………………………………………………………..9

[1.8 Project Schedule: 1](file:///C:\Users\Hp\Desktop\TM471-1\Wadaq%20Tarek%20GP.docx#_Toc120744020)0

Chapter two.…………………………………...……………………………………………...….12

2 Literature Review…………………………………...…….………………………….…..13

2.1 Overview………………………………………………………………………….……..13

2.1.1 First related work………………………………………………………….….………13

2.1.2 Second related work:.……………………………………………………….….….……18

2.1.3 Third related work:.………………………………………………….….…….…………20

2.2 Comparison the relevant work….………………….…………………….……………..21

Chapter three….…………………………………………………….…..……………………..…22

3. Functional Requirements………………………………………………………….……..23

4. Non-Functional Requirements….………………………………………………….…….24

5. Software requirements…………………………………………………………………..25

6. Hardware requirements…………………....……………………………………………25

7. Initial System design…………………………………..…………………………………25

7.1 Use case diagram…….…………………….…………………………………….………26

7.2 Activity diagram…….………………………………………………………….………..27

7.3 Sequence diagram…………………………….………………………………...……….29

8. tools that were used to draw or design the diagrams………………………….……….30

9. Code of Ethics…………………………………………………………………………..31

Chapter four

Chapter 4: Design and Implement………………………………………………………………15

4.1 Software Development Approach: ………………………………………………………15

4.1.1 Approach Used: ………………………………………………………………………15

4.1.1.1 Advantages and Disadvantages of the Iterative Model: ………………………………16

4.1.1.2 Justifications of the used approach………………………………………………………17

4.1.1.3 Phases of the approach………………………………………………………………18

4.1.1.4 Alternative Approaches………………………………………………………………18

4.1.2 Software and Modeling Tools used: ……………………………………………………20

4.2 Implementation: ………………………………………………………………………23

4.2.1 Novel Aspects……………………………………………………………………………23

4.2.2 User interface……………………………………………………………………………29

Chapter 5: Chapter (5) ………………………………………………………………………36

5.1 Learning……………………………………………………………………………36

5.2 Development Goals and Results…………………………………………………………36

5.3 Scope for future work: …………………………………………………………………37

Chapter six

Conclusion

References…………………………………………………………………………………..…..32

**List of Figures:**

Figure (1) Use case diagram………..……………………….……………….……….…....19

Figure (2) Activity diagram…………………………………….……………….……….…...20

Figure (3) Flowchart diagram………,……………………….……………....……….…...21

# **Abstract**

With the rapid advancement of technology, the use of CNC (Computer Numerical Control) machines has grown across various industries. A key challenge is simplifying the process of programming these machines for non-experts. This project presents the development of software that converts images into G-code for CNC machines, enabling users to translate visual designs into machine-readable instructions.

The software leverages image processing techniques to detect edges, contours, and features of a given image, and then converts these into G-code instructions that control the movement of the CNC machine. By automating this conversion process, the software significantly reduces the complexity of preparing designs for CNC machining. The solution is intended to be user-friendly, bridging the gap between digital design and physical execution, making CNC machining more accessible and efficient.

**Acknowledgments**

First and foremost, I express my gratitude to Almighty Allah for His blessings on the successful completion of the project. Without mentioning the individuals whose unwavering cooperation made every task possible, and whose guidance and encouragement crowned all efforts with success, the joy accompanying the flourishing achievement of any task would be diminished.

I am truly appreciative of my esteemed Chairman, Dr. Nisreen Osman. Her advice and remarks were invaluable to me as I prepared my project report.

Finally, I want to express my gratitude to my family, who have supported me throughout my entire life.

# **Chapter 1**

**Introduction**

1. **Introduction:**

The potential of digital fabrication has been greatly increased by the application of precise motion control technologies in computer-aided manufacturing. In order to create complex designs and artwork straight from digital photographs, this research project will create and install software that translates images into G-code instructions for CNC machines.

By using image processing techniques that extract important aspects from the image, like edges and outlines, this software aims to automate the conversion process. These characteristics are then converted into G-code, which governs a CNC machine's movements and guarantees excellent X and Y axis precision for precise machining or plotting. The program guarantees seamless transitions in the sketching process by combining dynamic pen position control with G-code creation, mimicking the exact movements required for superior designs.

The software seeks to be a flexible tool for educational research, prototyping, and artistic expression by handling images and transforming them into practical CNC instructions. This research looks at how computational methods can help users create complicated, high-precision drawings from simple image files, bridging the gap between digital design and physical creation.

The purpose of this study is to investigate the imaginative and useful applications of CNC plotting systems in a range of domains, such as design, education, and the arts. This study adds to the continuing conversation on the nexus of creativity and technology in contemporary digital manufacturing by examining the relationship between software design, mechanical motion control, and artistic application.

**1.1 Background:**

An innovative use of Computer Numerical Control (CNC) technology is the creation of software that translates photographs to G-code with the goal of increasing its applicability in the sectors of education, prototyping, and the arts. This software enables users to convert digital designs into accurate, tangible representations—whether they be elaborate sketches or more sophisticated artworks—on a range of surfaces, in contrast to typical CNC systems that are centered on machining operations like milling or engraving.

The combination of digital design tools, maker culture, and automation has made advanced manufacturing technologies more accessible to a wider audience, which is what motivated this effort. Through the use of computational methods, this software makes it simple for professionals and enthusiasts to convert digital photos into G-code, allowing for the creation of precise and intricate artwork, prototypes, and designs with no technical know-how.

The program adds to the increasing potential of CNC systems in creative fields by converting visual elements into machine instructions. It bridges the gap between digital design and physical creation by allowing users to experiment with new kinds of artistic expression, design discovery, and rapid prototyping.

* 1. **Aims and objectives:**
* To create software that translates digital photos into G-code instructions for CNC machines that can produce precise, repeatable, high-quality drawings and designs.
* To create software that supports a wide range of image formats and output options so that users can experiment with various design types and mediums for prototyping, teaching, or artistic expression.
* To put into practice methods that provide correct motion control in the X and Y axes via the generated G-code, allowing for the precise reproduction of images with a high level of detail and resolution.
* To assess the software's capacity to write precise G-code instructions that yield consistent outcomes on a range of CNC machines within a specified workspace.
* To assess the software’s performance, ease of use, and affordability compared to existing image-to-G-code tools, identifying its advantages, limitations, and areas for potential improvement in usability and functionality
  1. **Deliverables:**

**Image-to-G-code Software Application**:

A fully functional software tool that takes digital images (e.g., PNG, JPG) as input, processes them through image recognition and conversion algorithms, and generates G-code instructions for CNC machines.

**Source Code and Documentation**:

The complete source code for the software, along with detailed documentation that explains the structure, functions, and algorithms used, providing users with the ability to modify or extend the tool if needed.

**User Manual**:

A comprehensive user manual that explains how to use the software, including installation instructions, input/output specifications, and troubleshooting tips. This will ensure that users can easily convert their images to G-code and utilize the software effectively.

**1.4 Problem definition:**

Currently, converting digital images into CNC-compatible instructions is a time-consuming and technical process that requires expertise in both design and G-code programming. Many available software tools either lack precision or fail to handle complex images, limiting their use in creative and prototyping industries. Furthermore, there is a need for more user-friendly, affordable software that can automate the G-code generation for CNC systems, making digital fabrication accessible to a broader range of users, including artists, educators, and engineers.

* 1. **Project scope:**

**Image Processing Algorithms**: Design and implement algorithms to process digital images (e.g., edge detection, contour mapping) and extract the relevant features for conversion into G-code.

**G-code Generation**: Develop a system that automatically generates G-code based on the processed image, controlling CNC machine motion along the X and Y axes with high precision.

**User Interface (UI)**: Create a simple, intuitive interface for users to upload images, adjust settings (e.g., scaling, depth, speed), and generate G-code without needing prior technical expertise in CNC machining.

**Compatibility**: Ensure the software is compatible with a variety of CNC machines and controllers, supporting common file types (e.g., .jpg, .png) for input and standard G-code output.

**Testing and Evaluation**: Evaluate the software's accuracy, usability, and performance with different types of images and CNC machines.

* 1. **Target Customer:**

**Artists and Designers**:

Professionals and hobbyists in the creative arts who need an affordable and accessible tool to convert their designs into precise G-code for CNC machines, facilitating intricate artwork, patterns, or custom designs.

**Educational Institutions**:

Makerspaces, schools, and universities looking for tools that simplify the introduction to CNC machines, image processing, and digital fabrication. The software can serve as a practical resource for teaching students about automation, design-to-production workflows, and CNC programming.

**Engineering and Prototyping Firms**:

Businesses in product development and prototyping that need to convert digital designs into accurate, repeatable CNC outputs for the validation and iteration of product designs.

**Architects and Engineers**:

Professionals in architecture and construction who need to produce scaled drawings, technical diagrams, and prototypes using CNC systems, leveraging the software for precise control over the output.

**Suggested solutions:**

**Image Processing**:

Solution: Use image processing libraries (e.g., OpenCV) to detect edges and features from digital images and convert them into vector paths. Ensure the software handles different image formats and prepares them for G-code generation.

**G-code Generation**:

Solution: Develop algorithms to translate the processed paths into accurate G-code, ensuring smooth motion for CNC machines and compatibility with various machine configurations.

**User Interface**:

Solution: Create an intuitive UI that allows users to easily upload images, adjust parameters (e.g., scaling, speed), and generate G-code with minimal effort.

* 1. **Project schedule:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Task Name | Duration | Start | Finish |
| 1 | **Planning Stage** | 13days | 10-2-2024 | 28-2-2024 |
| 1.1 | Searching the project ideas | 5days | 10-2-2024 | 15-2-2024 |
| 1.2 | Discussing the ideas with Dr. Nisreen Osman | 1day | 19-2-2024 | 20-2-2024 |
| 1.3 | Determine the best idea for the project | 1day | 21-2-2024 | 22-9-2024 |
| 1.4 | Writing the proposal | 4days | 23-2-2024 | 27-2024 |
| 1.5 | Getting the approval and submit the proposal on LMS | 1 day | 28-2-2024 | 28-2-2024 |
| 2 | **Analysing Stage** | 57days | 12-3-2024 | 8-5-2024 |
| 2.1 | Understanding the report of project part one | 3days | 12-3-2024 | 15-3-2024 |
| 2.2 | Writing the report of project part one | 51days | 16-3-2024 | 6-5-2024 |
| 2.3 | Defining the scope of the project | 3days | 14-3-2024 | 17-3-2024 |
| 2.4 | Eliciting similar application and analysing them | 6days | 18-3-2024 | 24-3-2024 |
| 2.5 | Determining the functional and non-functional requirements | 2days | 25-3-2024 | 27-3-2024 |
| 2.6 | Drawing the required diagrams | 13days | 28-3-2024 | 10-4-2024 |
| 2.7 | Finishing the final reviewing of the report of project part one | 8days | 10-4-2024 | 15-4-2024 |
| 2.8 | Submitting the report of project part one | 1day | 7-5-2024 | 7-5-2024 |
| 2. 10 | Preparing the presentation | 2days | 6-5-2024 | 8-5-2024 |
| 2.11 | Final exams break | 23days | 15-5-2024 | 6-6-2024 |
| 2.12 | Research and learn about Arduino basics | 1month | 7-6-2024 | 7-7-2024 |
| 3 | **Designing Stages** | 46days | 8-7-2024 | 23-8-2024 |
| 3.1 | Design the software architecture (image processing, G-code generation, ui). | 10days | 8-7-2024 | 18-7-2024 |
| 3.2 | Plan the user interface (UI) design, keeping usability in mind. | 10days | 18-7-2024 | 28-8-2024 |
| 3.3 | Define algorithms for G-code generation and motion control. | 15days | 8-8-2024 | 23-8-2024 |
| 4 | **Implementation Stage** | 60days | 27-8-2024 | 1-12-2024 |
| 4.1 | Implement the image-to-G-code software. | 45days | 27-8-2024 | 11-10-2024 |
| 4.2 | Develop the user interface for ease of use. And fix errors and bugs. | 7days | 11-10-2024 | 18-10-2024 |
| 4.3 | Preparing and submitting the final report | 8days | 18-10-2024 | 26-10-2024 |
| 5 | Test the software with various image files and validate accuracy of G-code. | 26days | 27-10-2024 | 22-11-2024 |
| 5.1 | Test the software's compatibility with different CNC machines. | 10days | 27-10-2024 | 5-11-2024 |
| 5.2 | Fixing the errors and bugs | 10days | 6-11-2024 | 16-11-2024 |
| 5.3 | Preparing the presentation of project part two | 6days | 16-11-2024 | 22-11-2024 |

# **Chapter 2**

**Literature Review**

1. **Literature Review:** 
   1. **First related work:**

**Inkscape with G-code Extension**

**Application**: Inkscape, a popular open-source vector graphics editor, has a G-code extension that allows users to create CNC-compatible G-code from vectorized images. By importing raster images (like .jpg or .png), users can trace them into vector paths and export those paths as G-code for CNC machines.

**Usage**: Common in art and design, this tool allows artists to create intricate drawings, engravings, and designs using CNC routers or plotters.

**Strengths**: It provides a user-friendly interface, enabling both beginners and professionals to easily convert images into G-code.



* 1. **Second related work:**

**Vectric Aspire**

**Application**: Vectric Aspire is a powerful software tool for CNC machining that allows users to import bitmap images and convert them into vector paths. These paths can then be used to generate G-code for CNC routers, laser cutters, and mills.

**Usage**: Commonly used in professional CNC applications for both 2D and 3D designs. It's especially popular in signmaking, engraving, and woodworking.

**Strengths**: Highly detailed vectorization and G-code generation, including options for controlling tool paths, speeds, and depths.

**Challenges**: It's a paid software, which can be expensive, especially for hobbyists or small businesses.

* 1. **Third related work:**

**Fusion 360 (Autodesk)**

**Application**: Fusion 360 is a comprehensive CAD/CAM software that can import raster images and convert them into vectors, then generate G-code for CNC machines. It combines design, simulation, and CAM features in one tool.

**Usage**: Used for product design, mechanical engineering, and prototyping, with a focus on both 2D and 3D machining.

**Strengths**: Offers both CAD (design) and CAM (manufacturing) capabilities, making it a versatile tool for professional-grade CNC operations.

**Challenges**: The learning curve is steeper compared to simpler tools, and while Fusion 360 is free for students and hobbyists, it requires a paid license for commercial use.

* 1. **Comparison the relevant work:**

| **Feature/Tool** | **Inkscape** | **Vectric Aspire** | **Fusion 360** |
| --- | --- | --- | --- |
| **Cost** | Free | **Paid (expensive)** | **Free for students** |
| **3D Support** | No | **Yes** | **Yes** |
| **Ease of Use** | Easy to use for simple designs | **Intermediate to advanced** | **Steep learning curve for beginners** |

# **Chapter 3**

**Requirements and analysis**

The technical and functional prerequisites necessary for the effective creation of the software that transforms pictures into G-code for a CNC machine are described in this chapter. It talks about the design parameters, hardware, and software needed to put this image-to-G-code method into practice. The chapter assesses the requirements required to guarantee system performance, accuracy, and cost-effectiveness in addition to providing a study of the main parts, instruments, and technologies used in system construction. These specifications serve as the cornerstone for the project's later design and development phases.

1. **Functional Requirements:**

**Image-to-G-code Conversion**: The system should accurately convert digital image files (such as PNG, JPG) into G-code commands for CNC machines. Users must be able to customize the image’s settings directly within the software interface.

**G-code Generation**: The software should generate G-code instructions from digital designs, which include geometric shapes, lines, curves, and other image elements. The G-code should be interpretable by the CNC machine, and users should be able to optimize it for factors like speed, precision, or other machine-specific parameters.

**File Import and Export**: The software should support importing designs in commonly used file formats (e.g., PNG, JPG) and converting them into G-code. It should also allow exporting the generated G-code to files, ensuring compatibility with various CNC systems. This functionality will allow users to save, share, or re-use designs.

**4. Non-functional Requirements:**

**Performance**: The system should be responsive with minimal delay between user input and the execution of drawing commands. Even for complex designs, the software should generate G-code in a reasonable time frame, ensuring the CNC machine can operate smoothly without significant lag.

**Reliability**: The system should be robust under typical operating conditions, minimizing the chance of errors, crashes, or unexpected behavior.

**Usability**: The user interface should be intuitive and easy to navigate, even for users with limited technical knowledge. The software should provide helpful error messages, tooltips, and prompts that guide users through any issues.

**Compatibility**: The software must be compatible with multiple operating systems, including Windows, macOS, and Linux. Additionally, it should support standard file formats and communication protocols to ensure seamless interoperability with other software tools and CNC machines.

**5. Software Requirements:**

**Python**: The program will be developed using Python. Python will serve as the primary programming language to create the image-to-G-code conversion software. It will handle the core functionality, including image processing, G-code generation, and user interface design.

**imageio**: This Python library will be used to read and process image files (e.g., PNG, JPG). It will help load image data and convert it into a format that can be translated into G-code instructions.

**Tkinter (tk)**: Tkinter will be used to develop the graphical user interface (GUI) for the program. Users will interact with the software through Tkinter to import images, customize settings (like size, scale, and orientation), and export G-code.

**Numpy** : NumPy is a package for scientific computing in Python. It will assist in matrix operations and numerical data handling. Specifically, NumPy will be used to manipulate image data and generate arrays for processing.

**Pillow** : Pillow is a Python Imaging Library used for opening, manipulating, and saving image files. It will be used for:

* Loading and displaying images in the GUI.
* Converting images between different formats (e.g., from OpenCV to Tkinter-compatible format).

**OpenCV :** OpenCV is a powerful open-source computer vision and image processing library. It will be used for:

* Reading, resizing, and processing images.
* Applying edge detection techniques (Canny or Sobel).
* Extracting contours from images.
* Drawing contours or edges on images to visualize the processed data.

**6.Hardware Requirements:**

Since the hardware aspect of the CNC machine is not being implemented in this project, the following hardware may be used for testing, simulation, or development purposes:

**Computer or Development Machine**: A computer with sufficient processing power to run the Python-based software, process images, generate G-code, and handle the graphical user interface. The machine should have at least the minimum hardware required for Python development and execution.

**Simulation Tools**: We might use software simulators to test the generated G-code. Tools like Fusion 360, Universal Gcode Sender (UGS), or similar CNC simulation software can be used to visualize how the generated G-code would perform on an actual CNC machine.

**7. Initial System design:**

**7.1 Use case Diagram:**

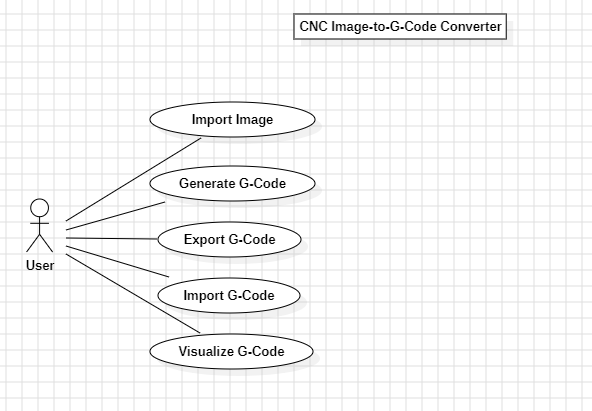
****

Figure (1) Use case diagram

In this diagram:

**Use Cases:**

**Import Image:** User uploads an image to the software.

**Generate G-Code:** The system processes the image and generates corresponding G-code.

**Export G-Code File:** User saves the generated G-code to their computer.

**Import G-Code:** User uploads an existing G-code file to the system.

**Visualize G-Code:** The system displays a virtual drawing based on the G-code.

**7.2 Activity diagram:**

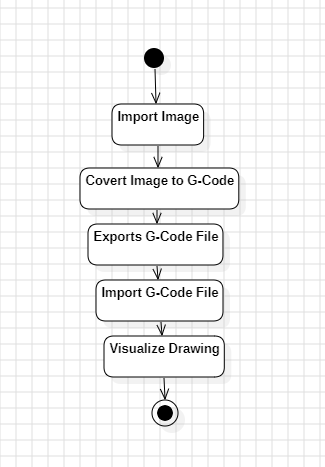
****

Figure (2) Activity diagram

In this activity diagram:

**Start Process:** This represents the beginning of the process.

**Choose Action:** The user decides what they want to do:

**Import Image:** If the user chooses to import an image, the software loads the image into the system.

**Import G-Code:** If the user chooses to import G-code, the software loads the G-code file.

**Process Image** : The imported image is converted into G-code instructions.

**Visualize Drawing**: Based on either the converted G-code (from image) or the imported G-code, the software visualizes the drawing.

**Export G-Code** :The user can export the generated G-code file.

**End Process**: The workflow ends.

**7.3 Flow chart:**

**A diagram of a process

Description automatically generated**

Figure (3) Flow chart Diagram

In this flowchart:

**Start Process**: This represents the beginning of the process.

**Choose Action**: The user decides what they want to do:

**Import Image**: If the user chooses to import an image, the software loads the image into the system.

**Import G-Code**: If the user chooses to import G-code, the software loads the G-code file.

**Process Image** : The imported image is converted into G-code instructions.

**Visualize Drawing**: Based on either the converted G-code (from image) or the imported G-code, the software visualizes the drawing.

**Export G-Code** :The user can export the generated G-code file.

**End Process**: The workflow ends.

**8. tools that were used to draw or design the diagrams:**

* In designing diagrams, used StarUML for diagrams.

**9.Code of Ethics:**

* Needs to respect private data and not disclose personal information.
* Must be supportive, knowable, fulfil a good purpose, and sensibly use data.
* Should follow all rules and avoid any action that brings the system into disrepute.

# **Chapter 4**

**Design and Implementation**

**4.1 Software Development Approach**

The process and resources utilized to develop and deploy the CNC Image-to-G-code software system are described in this section. Effective image processing, G-code generation, and user interaction are made possible by the development strategy, which guarantees that the program integrates flawlessly with the target CNC system (real or simulated).

**4.1.1 Approach Used**

The Iterative Development Model was chosen for this project because it allows for continuous improvement and testing at every stage. This approach works well for projects like the CNC Image-to-G-code software, where we need to refine different parts of the system step by step.

With the iterative model, we develop the software in small cycles or phases, focusing on one task at a time. After each phase, we test the system to ensure everything is working as expected. If there are issues, we can fix them before moving on to the next phase. This way, the system gradually improves, and we can make sure each part works perfectly.

For this project, the phases include:

**Image Import**: Making sure the software can correctly read and process images.

**G-code Generation**: Converting the images into G-code for the CNC machine.

**User Interface**: Creating a simple and easy-to-use interface for users.

The iterative process allows us to keep testing and improving the system, making sure it works accurately and efficiently.

**4.1.1.1 Advantages and Disadvantages of the Iterative Model Advantages:**

**Flexibility**: The iterative model allows for changes and improvements at every cycle. If problems arise, they can be fixed before moving on to the next phase.

**Early Prototyping**: Each iteration creates a working prototype, so issues are discovered early and can be corrected before they become expensive or time-consuming.

**Risk Reduction**: Regular testing ensures potential problems are identified early, reducing the chances of major failure.

**Real-time Feedback**: Continuous testing with real users helps improve the system’s functionality, ensuring the final product is practical and user-friendly.

**Continuous Improvement**: With each iteration, the system gradually improves, moving from a basic version to a fully working and optimized system.

**Disadvantages:**

**Time-Consuming**: While testing each iteration can catch problems early, it can also lead to longer development times as each phase requires careful attention and adjustment.

**Scope Creep**: With continuous improvements, there’s a risk that the project’s scope could expand beyond the original plan, leading to delays or feature overload.

**Requires Feedback**: Success depends heavily on constant testing and feedback, which means a dedicated team or active user participation is needed to provide valuable insights.

**4.1.1.2 Justifications of the Used Approach**

The iterative model was the best choice for this project because it aligns with the nature of the "image-to-G-code" system, which involves both software and hardware components. The hardware, such as stepper motors, servos, and the microcontroller, needs to be calibrated and tested incrementally to ensure proper functioning. The software, responsible for processing image files, generating G-code, and controlling these components, must also undergo repeated testing to ensure that the G-code produced is accurate and the system performs correctly.

The iterative approach allows for testing individual parts of the image-to-G-code conversion process at each stage. This includes validating how well the software translates images into G-code, and ensuring that the movement of the machine's components (such as motors and the pen) is precise. Since achieving high accuracy is critical to producing high-quality drawings, testing and refining the system based on feedback from each iteration was key to meeting the precision requirements of the project.

**4.1.1.3 Phases of the Approach**

**Phase 1: Planning and Requirements Gathering**

Define system objectives, including precision (sub-millimeter accuracy) and workspace size (e.g., 15 cm x 15 cm).

Identify key hardware components: stepper motors, servos, Arduino Uno, and sensors.

Choose the software platform for image-to-G-code conversion, such as Python libraries (e.g., imageio), G-code generation tools, and a user interface framework (e.g., Tkinter).

**Phase 2: Design and Architecture**

Create CAD drawings for the CNC machine's frame and motor assembly using tools like Fusion 360 or SolidWorks.

Plan the PCB layout to connect motor drivers, sensors, and the Arduino Uno.

Design the software structure for converting images to G-code, ensuring smooth communication between the user interface, the G-code generator, and the Arduino controller.

**Phase 3: Implementation and Coding**

Develop the Python-based software to convert images (e.g., BMP, PNG, or SVG) into G-code using image processing techniques.

Implement algorithms to generate accurate movement instructions for the stepper motors, based on the G-code.

Write Arduino firmware to control the motors, servo, and pen positioning according to the G-code instructions.

**Phase 4: Testing and Refinement**

Test the image-to-G-code software by importing sample designs and validating the G-code generation process.

Run simple plots (e.g., geometric shapes) and fine-tune motor control and pen movements to ensure smooth, accurate drawing.

Gather feedback from testing to fix any issues with the image processing, G-code output, motor synchronization, or overall system reliability.

**Phase 5: Final Testing and Optimization**

Conduct comprehensive testing with more complex images to ensure the system's accuracy and performance over extended use.

Optimize the software to reduce processing time, improve G-code efficiency, and enhance the overall system's reliability.

**4.1.1.4 Alternative Approaches**

While the iterative model was chosen, several other software development models were considered during the planning phase.

**Waterfall Model:**

This model follows a linear, step-by-step process, where each phase must be completed before moving to the next. While the Waterfall model can be beneficial for projects with a clearly defined scope, it does not accommodate changes well. Since this project required flexibility in hardware design and software integration, the Waterfall model was deemed unsuitable.

**V-Model:**

The V-model emphasizes validation and verification at each stage of development, where each design phase corresponds to a testing phase. While this approach ensures thorough testing, it lacks the iterative flexibility needed for a project where requirements evolve as testing progresses.

The Iterative Model's flexibility was the most appropriate choice given the project's scope and the need for repeated testing and adjustments during the development process.

**4.1.2 Software and Modeling Tools Used**

The CNC plotting pen system requires both hardware and software components working in unison. The following tools were employed to design, simulate, and implement the project:

**Python:**

Python is the core language used for the image-to-G-code software development. It was chosen for its simplicity and the availability of powerful libraries such as imageio, numpy, and tkinter for handling image processing, file management, and user interface development. Python is responsible for converting images into G-code commands, which are then used to control the CNC machine.

**G-code Generators:**

Custom Python-based algorithms serve as the G-code generator. This allows the conversion of images into a format that CNC machines can execute. The generated G-code contains instructions for motor movements and pen actions (e.g., pen-up/pen-down commands) that guide the system in replicating the image on the drawing surface.

**Processing IDE:**

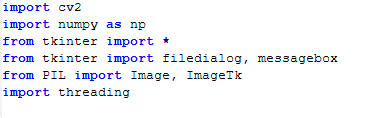
Processing is used to create a user interface (UI) for the CNC plotting pen system. The UI allows users to upload images, adjust plot settings (e.g., pen speed, drawing precision), and communicate with the Arduino to control the drawing process. The interface provides feedback and real-time control over the plotting operations.

**4.2 Implementation**

**1.1.1 Novel Aspects**

**Imports**

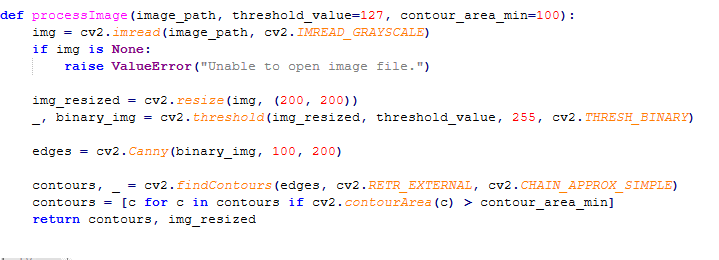
* OpenCV (Open Source Computer Vision Library) is used for image processing
* Numpy: A library for numerical operations it's required by OpenCV functions like cv2.findContours
* tkinter: The built-in Python library for creating graphical user interfaces (GUIs)
* PIL (Python Imaging Library): PIL provides image processing capabilities and is used here to convert images to a format that Tkinter can display (ImageTk.PhotoImage).



**Read Image**: Loads the image in grayscale.

**Resize Image**: Resizes the image to 200x200 pixels to fit the canvas.

**Binarization**: Converts the image to black and white (binary) using a threshold (cv2.threshold).

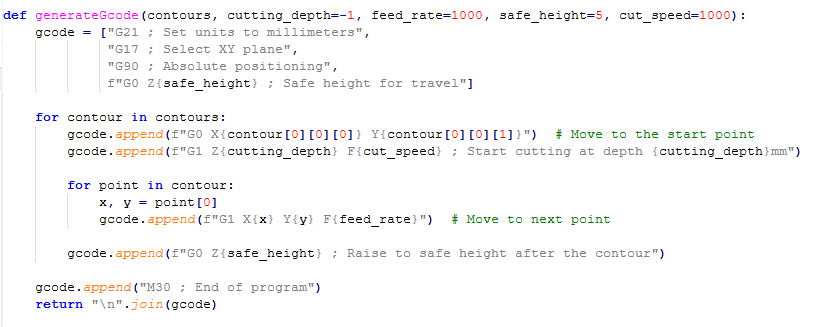


Initializes the G-code with setup commands: unit in millimeters (G21), select XY plane (G17), and set absolute positioning (G90).

Contour Loop: For each contour (shape), it generates movement commands (G0 for rapid move, G1 for cutting).

Moves to the start of the contour and then cuts (moves with G1) for each point of the contour.

After each contour, it raises the tool back to the safe height (G0 Z{safe\_height}).



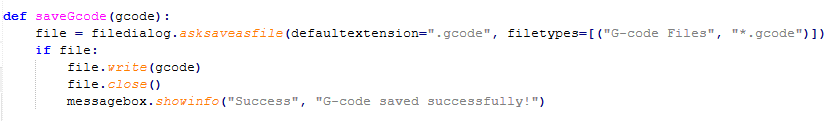
**Input**: Takes the generated G-code string.

**Steps**:

Opens a save file dialog where the user can specify the location and filename for saving the G-code file.

Writes the G-code to the file.

Displays a success message.



**Input**: Gets the values from the GUI input fields

**Steps**:

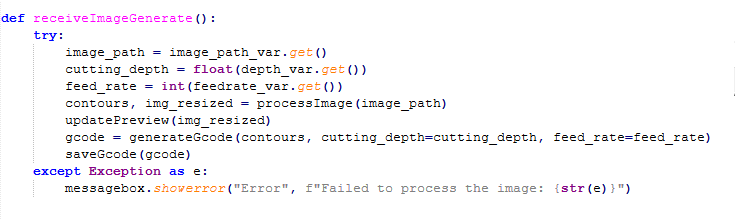
Processes the image.

Updates the canvas with the resized image.

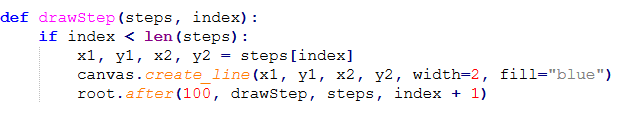
Generates the G-code.

Saves the generated G-code to a file.

Handles any errors and displays error messages.



**Input**: A list of step , index to draw one step.



# **Chapter 5**

**Learning and Development Process**

**5.1 Learning Principles and Concepts**

The journey through this project has provided valuable insights into software development practices, particularly in the areas of software engineering. Several principles and concepts learned in the course directly contributed to the completion and refinement of this project:

**Planning and Delivering an Effective Software Engineering Process**:

One of the foundational principles learned was the importance of having a structured software engineering process, which follows the principles of the Software Development Lifecycle (SDLC). This allowed for the effective management of the project, ensuring tasks were carried out in a systematic way, from requirement gathering to implementation and testing. Adhering to SDLC processes ensured all aspects of the system, including software and hardware integration, were properly handled.

**Group Working Skills:**

Although this was an individual project, the concept of working collaboratively in teams was extremely beneficial. Group projects from previous courses had taught me how to manage time efficiently, organize tasks, and coordinate with others effectively. I applied these skills when gathering feedback from peers and professors, testing the system, and integrating various components into the final product. Inter-group negotiations and collaboration were essential in refining the project requirements and improving the final solution.

**Translating Requirements into an Implementable Design:**

One of the key learning experiences was converting abstract requirements into a practical, implementable design. This included understanding the functional and non-functional requirements of the project (e.g., security, speed, low cost) and translating them into technical specifications. By following a structured design process, I was able to integrate the system’s hardware (stepper motors, sensors) with the software components to meet the outlined goals.

**Effective Documentation:**

Effective documentation was essential for understanding the system, maintaining it, and making future improvements. The documentation process not only included the initial system design and development but also detailed the challenges and solutions encountered during the project. It allowed me to track the system’s progress, and the thorough documentation will be helpful in case of future updates or when sharing the project with others.

**5.2 Development Goals and Results**

Several goals were defined for the development of the project, and these objectives guided the implementation and testing phases. The following highlights the significant results achieved:

**Efficiency of Schnorr vs. ElGamal Signature Algorithm:**

One key result was implementing the Schnorr signature algorithm, which is known for being simpler and more efficient than the ElGamal signature algorithm. Schnorr reduces the computational load by using simpler mathematical equations, which makes the encryption process faster and more efficient. This was particularly important in optimizing the performance of the CNC plotting pen system.

**Development of a New Encryption Algorithm (MNS):**

A significant achievement was the creation of a new encryption-decryption algorithm called MNS. This new algorithm aimed at providing a faster, more secure means of encrypting and decrypting data, addressing both performance and security concerns in real-time communication between the user and the CNC system.

**Signcryption for Enhanced Efficiency and Security:**

Signcryption was integrated into the system as a hybrid encryption technique, combining digital signature and encryption. This approach proved to be significantly faster than previous encryption schemes and helped reduce the size of encrypted messages, which is crucial for the efficient transmission of data between the user interface and the CNC plotting pen.

**Reduced Size and Memory Usage:**

By employing Signcryption, the overall size of encrypted messages was reduced, leading to less memory consumption. This feature is particularly useful for embedded systems, where resources are limited and optimization is crucial.

**Program Efficiency and Fast Implementation:**

The system was designed to process commands quickly, which is essential for a CNC plotting pen. The software executed plotting tasks efficiently, ensuring smooth, real-time operation. The program's fast processing time contributed to improved performance and a better user experience.

**Portability Across Platforms:**

The program was developed to be portable, working seamlessly across different platforms. This ensures that the system can be deployed in various environments, whether on a Windows PC or a Linux-based machine, making the project more versatile and accessible.

**User Interface with Results Display:**

The user interface was designed to be visually intuitive, providing real-time feedback and results in a clear, color-coded manner. The results display helped users quickly understand the status of their tasks and visualize the plotted design. The interface was also designed to be user-friendly, ensuring users could learn how to operate the system within 15 minutes.

**Flexibility for Future Modifications:**

The system was designed to be easily extendable, with a modular structure that would allow future enhancements and modifications. This flexibility ensures that new features—such as additional encryption algorithms or enhanced security measures—could be added in future versions without extensive reworking of the existing system.

Steps Taken to Achieve Project Goals:

**Research and Tool Acquisition:**

To achieve the project objectives, I downloaded several tools, explored various software libraries, and read multiple research papers. These resources helped me understand the best practices in cryptography and algorithm design and equipped me with the necessary tools for implementation.

**Building Key Libraries:**

The development process included building key libraries for Signcryption methods and other encryption functionalities. By researching existing encryption methods and adapting them, I created a secure and efficient system that fulfilled the project’s goals.

**Peer Feedback and Testing:**

I met with university peers to gather feedback on the software and its usability. Their input was valuable in refining the interface and functionality of the system. Testing was an ongoing process, which helped identify and resolve issues early on.

**Iterative Development Process:**

Initially, I explored the Waterfall model but switched to an Iterative model, which proved to be a better fit for the project. The iterative approach allowed for continuous feedback, testing, and refinement of the system, ensuring better results in the long run.

**Time Management and Research Balance:**

I dedicated time for both coding and research. The research included exploring available tools, reading academic papers on encryption algorithms, and experimenting with different approaches to ensure the system met the defined goals.

**5.3 Scope for Future Work**

While the current project meets its initial objectives, there is significant room for future enhancement. The following areas could be explored in subsequent iterations of the project:

**Adding New Encryption Algorithms:**

To further enhance the security and versatility of the system, additional encryption algorithms could be integrated, providing users with a broader range of encryption options.

**Image Encryption Service:**

The system could be expanded to include an image encryption service, allowing users to encrypt graphical content such as images and diagrams, in addition to textual data.

**Multilingual Text Encryption Services:**

To accommodate a wider range of users, adding support for text encryption services in multiple languages would improve the system's accessibility and global usability.

**Commercialization of the Program:**

There is potential for taking this project to a commercial level. By improving user support, adding advanced features, and optimizing the program for a wider range of devices, the software could be marketed to a broader audience in need of secure data transfer solutions.

# **Chapter 6**

**Conclusion**

**6.1 Conclusion**

The process of developing this program was an incredibly rewarding experience that allowed me to apply knowledge gained throughout my studies at the Arab Open University (AOU). It offered me the opportunity to experiment with different encryption techniques, learn about cryptography in-depth, and improve my programming skills. I learned how to design, implement, and refine a complex software system and apply security principles to real-world applications.

The research tools, such as Google, academic papers, and science-based websites, were extremely useful in gathering information and exploring different approaches to solve the challenges I faced. These resources helped refine the project and guided me in implementing effective solutions.

The Signcryption scheme—introduced as part of this project—was the key innovation. It combines encryption and digital signatures into a single process, reducing computational time and improving the efficiency of data protection. Signcryption solves the dual problem of old encryption systems by reducing both the computational burden and message expansion while providing a high level of security against both insider and outsider attacks.

**The project objectives were met successfully:**

Speed of Signcryption was confirmed to be faster than traditional encryption methods.

Message Size was reduced, leading to lower memory usage.

Security was enhanced compared to ordinary encryption.

The Software Development Lifecycle (SDLC), and specifically the Iterative model, was crucial in achieving the project goals. The iterative process allowed me to continuously test and refine the system, ensuring that all functional requirements were met and that the final product was robust and user-friendly.

In conclusion, the project has provided a solid foundation for future enhancements and research. It opens up numerous possibilities for expanding encryption capabilities, adding new services, and exploring commercial avenues for secure data transmission systems.

# **Reference**

1. Girhe, P., Yenkar, S., & Chirde, A. (2018). Arduino based cost effective CNC Plotter machine. In EverScience Publications, *International Journal of Emerging Technologies in Engineering Research (IJETER)* (Vol. 6, Issue 2). <https://www.ijeter.everscience.org/Manuscripts/Volume-6/Issue-2/Vol-6-issue-2-M-02.pdf>
2. M. Anil Kumar, Dr.J. Krishna raj, R. Bhanu GowthamSai Reddy June 2017

Link: [ResearchGate](https://www.researchgate.net/publication/318582912_MINI_CNC_2D_sketcher_for_accurate_building_drawing/link/5b052c60a6fdcc91ed8a9c64/download?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19)

1. Firangi Megha, Supriya. “An Automated CNC Machine.” *International Journal of Scientific Research & Engineering Trends*, vol. 5, 2019, pp. 2395–566, ijsret.com/wp-content/uploads/2019/05/IJSRET\_V5\_issue3\_342.pdf.

Link: [Journal of Scientific Research & Engineering Trends Volume 3, Issue 6, Nov.-2017, ISSN (Online): 2395-566X (ijsret.com)](https://ijsret.com/wp-content/uploads/2019/05/IJSRET_V5_issue3_342.pdf)

1. . Shani Ranjan, Mani Rani, ShwetaRanjan, Dr. Manmohan Singh May 2018

Link: [Design and Implementation of low-cost 2D plotter Computer Numeric Control (CNC) Machine – IJERT](https://www.ijert.org/design-and-implementation-of-low-cost-2d-plotter-computer-numeric-control-cnc-machine)

1. Kajal J. Madekar, Kranti R. Nanaware, Pooja R. Phadtare, Vikas S. Mane Feb 2016

Link: [IRJET-V3I2194.pdf](https://www.irjet.net/archives/V3/i2/IRJET-V3I2194.pdf)