# Digital Twin-based Real-Time FDM Additive Manufacturing Defect Detection

Sahaj Patel
*Institute for Robotics and Intelligent Machines*
*Georgia Institute of Technology*
Atlanta, USA

***Abstract*—Fused Deposition Modeling (FDM) is a widely used additive manufacturing technique, but its complexity leads to various failure modes that waste material, damage equipment, and pose safety risks. This work proposes a real-time defect detection system for FDM 3D printing by leveraging digital twin simulations. A high-fidelity digital twin is used to generate per-layer expected print results. Convolutional neural networks (CNNs) perform style transfer to map between simulation and real-world image domains. Siamese neural networks (SNNs) then compare the actual print images to the digital twin predictions to detect anomalies and defects. The proposed multi-stage deep learning pipeline enables early detection of failure modes such as filament jams, part dislodging, under-extrusion, layer shifting, and hardware issues. Experiments demonstrate the effectiveness of the approach in identifying print defects in real-time, allowing print jobs to be paused to avoid material waste and printer damage. The digital twin framework provides a scalable and generalizable solution for online quality monitoring in additive manufacturing.**

***Index Terms*—digital twin, FDM, additive manufacturing, fault detection, industry 4.0**

## I. Introduction

As the Industry 4.0 era becomes more commonplace in manufacturing, Digital Twins are becoming more and more crucial in pushing the boundaries of fault detection, especially for additive manufacturing. Current methods often fail to identify issues accurately under varied conditions, leading to waste and safety concerns [1], [2]. Some existing work for this problem includes rolling out generalized computer vision models that can only detect large-scale print failures, such as the Spaghetti Detective system [3]. Digital twins are slowly being introduced to AM but namely in the form of material quality monitoring, often too specific a solution for wide-scale use. However, recent advancements in digital twin ecosystems for AM, such as the one proposed by Pantelidakis et al., provide a promising foundation for more comprehensive defect detection solutions [4]. This project introduces a new approach using a Convolutional Neural Network (CNN)-based neural style transfer (NST) model to reduce the discrepancy between real-world print job images and their digital twin simulations as a method to improve defect detection accuracy and reliability [4], [5]. Exploring the use of different neural network architectures, such as VGG19, will serve as the foundation for this style transfer model, similar to the StyleFlow approach proposed by Abdal et al. [6], [7]. These networks have been chosen for their potential to normalize factors like lighting and background variations that can hinder defect detection classification models. Additionally, Siamese Neural Networks (SNNs) will be employed for similarity detection between the style-transferred real-world images and the digital twin simulations [8]. SNNs have been shown to be effective in comparing and identifying differences between image pairs which can reveal faults or outliers [9]. By applying these models to a dataset of 3D print progress images, the project aims to identify the most effective method for enhancing image consistency and thus, improving the quality of fault detection models at large in these 3D printing processes for future use to make using and deploying digital twins more accessible.

## II. Problem Statement

Fused Deposition Modeling (FDM) is a widely used additive manufacturing technique, but its process complexity leads to various failure modes that waste material, damage equipment, and pose safety risks. Defects such as filament jams, part dislodging, under-extrusion, layer shifting, and hardware issues cause filament waste, especially in large print jobs that can take over 30 hours. Failed prints become unrepairable and must be disposed of, wasting even more material. Damage to the printer itself is also a major concern. Most critically, some failure modes can lead to thermal runaway, creating serious safety hazards [2]. Existing solutions like filament runout sensors and the Spaghetti Detective system provide limited error detection capabilities [3]. There is a strong need for a more comprehensive, generalizable, and robust defect monitoring solution to improve print quality, reduce material waste, and ensure safe FDM printer operation.

## III. Methodology

The methodology for this project consisted of dataset creation and model training and validation. Figure 1 shows the framework of the underlying design of this fault detection pipeline.

As no existing digital twin software was suitable for this task, a significant portion of the work involved creating a digital twin corresponding to the physical twin hardware setup. The hardware used was a BambuLabs P1P 3D printer, retrofitted with LED chamber lights and a timelapse camera. All printing was performed using white Overture PETG 1.75mm filament. Figure 2 shows the digital and physical

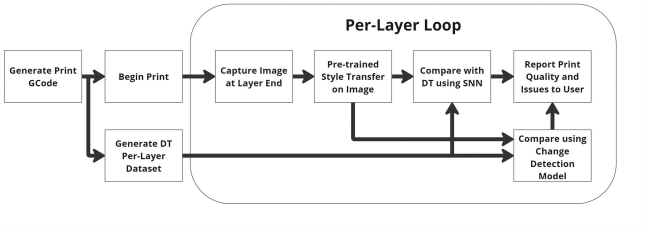


Fig. 1. Digital Twin-based framework for real-time fault detection.

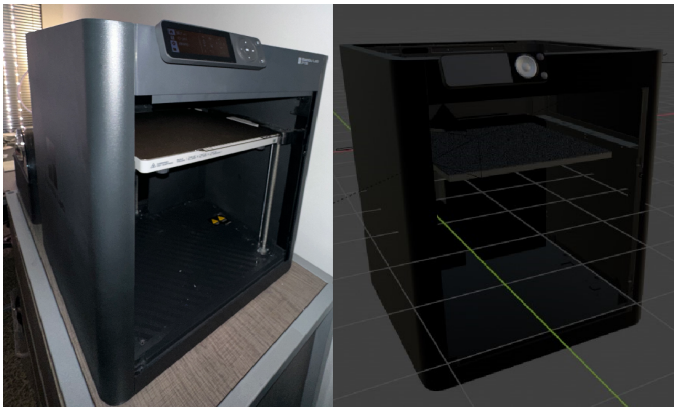

Fig. 2. Physical Twin (left). Digital Twin (right).

twins in the real world and the simulation environment, respectively.

### A. Digital Twin Dataset Generation

The digital twin simulation model was developed using Blender 3D animation and modeling software, chosen for its Python scripting capabilities and versatility [10]. 3D assets for the printer simulation models were acquired from the Bambu-Labs official website, open-source 3D model repositories, and custom-modeled in Fusion360. The models were imported into the Blender project, and a camera and lighting were added to closely match the physical twin's camera and LED lights. A script was developed to read the 3D printer's toolpath G-code file and convert the movement commands into 3D geometry, rendering it to visually resemble the physical twin's layer-by-layer print progress.

A diverse dataset of open-source 3D models was selected for printing, ranging from industrial parts with sharp edges to free-flowing artistic curvatures. The final models included a vase, a boat, an astronaut figurine, a rack-and-pinion gear system, and a lattice structure stress test. All models were configured for 0.2mm layer height, 0.4mm extrusion width, and 20% infill density, with other settings left as default in the BambuStudio slicer software used to generate the G-code files.

The physical twin's captured image dataset was created using the 3D printer's built-in timelapse capture functionality, enabled through the "Smooth Timelapse" option in BambuStudio. This setting captured images layer-by-layer, which were

Fig. 3. Captured image sample (left). Rendered image sample (right).

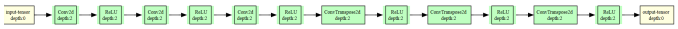

Fig. 4. Autoencoder geometry selected to train a NST model.

then stitched into a timelapse video at the end of each print job. The timelapse videos were split frame-by-frame to create the image dataset, with each frame corresponding to a layer of the print job.

For the digital twin, a Python script iterated through every layer in the G-code, generating the corresponding 3D geometry, translating the build platform to mimic the physical twin's print behavior, and rendering the scene from the camera. Bounding box polygon masks were created for each render and capture to remove non-essential background elements. Each captured image had a corresponding rendered image, with all images being 1280 by 720 pixels. The captures represent the real-world physical twin print progress, while the digital twin renders dataset represents the ground truth/expected results. Figure 3 shows samples of the captured and rendered datasets, where the camera placement and lighting conditions are similar.

### B. Neural Style Transfer

Two approaches were explored for neural style transfer: a custom autoencoder and a fine-tuned VGG19 model [11], [12]. This was to determine whether a small and computationally efficient model would be adequate or whether a very deep neural network would be required.

*1) Custom Autoencoder:* A custom autoencoder was developed and trained on the dataset to investigate its effectiveness in style transfer. The autoencoder architecture, as seen in Figure 4, consisted of an encoder and a decoder, with the encoder compressing the input image into a latent space representation and the decoder reconstructing the image from this representation. The autoencoder was trained using the Adam optimizer and mean squared error (MSE) loss function.

*2) Fine-Tuned VGG19 Model:* The pre-trained VGG19 model provided by PyTorch was fine-tuned for the dataset. Figure 5 depicts the VGG19 model architecture. The training process utilized a learning rate of 0.001, the Adam optimizer, and MSE loss for both the content and style loss. The content loss and style loss converged to values below 1e-6. Model validation for the style transfer approaches was primarily based on visual comparisons of the original image, the model's output, and the corresponding rendered image. The MSE loss between the model output and the target render image was used as a numerical benchmark, but it did not fully reflect the

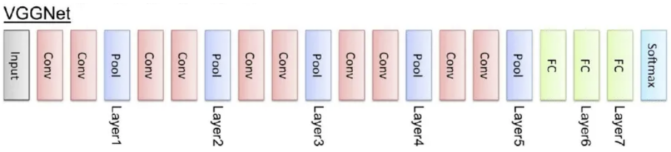


Fig. 5. Sample VGG19 model geometry. [13]

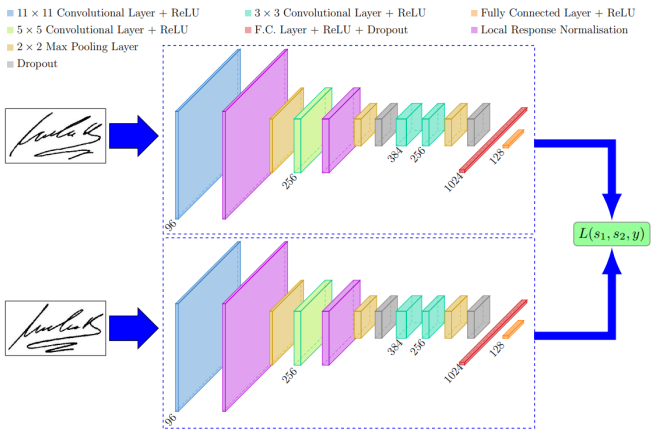


Fig. 6. SIGNet SNN model geometry. [15]

model's performance, as the outputs visually diverged from the expected results despite low loss values.

### C. Fault Detection

For fault detection, a Siamese Neural Network (SNN) model was developed, with a similar architecture to the one shown in Figure 6. The SNN consists of two identical subnetworks that share the same architecture and weights. The subnetworks are convolutional neural networks (CNNs) that extract features from the input images. The outputs of the subnetworks are then compared using a distance metric, such as Euclidean distance, to determine the similarity between the two input images. In this project, the contrastive loss function developed by Hadsell et al. was used for training the SNN model [14].

For fault detection, a Siamese Neural Network (SNN) model was developed and trained on 80% of the dataset used for style transfer. The style transfer results were used as a mask to remove the background from the input capture images during data preprocessing. The images were cropped to focus on the printed object and resized to 200 by 200 pixels to improve model training and avoid potential exploding gradients. The trained SNN model was validated using the remaining 20% of the dataset, and the resulting similarity score performance was recorded. The SNN architecture consisted of twin neural networks that processed the input capture image and the corresponding digital twin render, generating feature embeddings. The similarity between the embeddings was then computed to determine the presence of faults or anomalies.

## IV. Results

Given all the different approaches and methods tested in the development of this fault detection pipeline, the results are

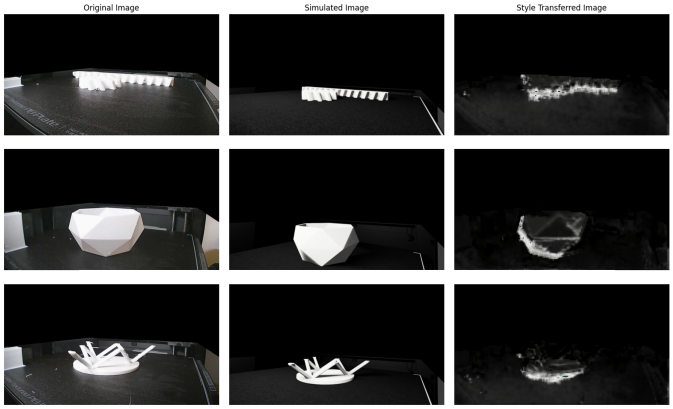


Fig. 7. Autoencoder Results on Test Dataset.

determined categorically for the dataset generation, the NST model, and the fault detection model.

### A. Digital Twin Dataset Generation

Dataset generation proved to be one of the most significant challenges in this project. The process involved creating a dataset for five classes of objects, including both real-world captures and simulated renders. The printing of the physical objects took 17 hours, while the generation of the simulated renders required 28 hours of computation time. It is important to note that these durations do not account for restarted print jobs necessitated by various issues such as print failures, power outages, computer hardware errors, and other related factors. The dataset generation process ultimately yielded 1,650 images of each type (captures and renders) across the five object classes, totaling 3,300 images. This extensive dataset generation effort was crucial for training and validating the proposed defect detection models, ensuring their robustness and generalizability across different object geometries and potential failure modes.

### B. Neural Style Transfer

The results for the NST exploration were divided into the autoencoder approach and the fine-tuned VGG19 model approach.

*1) Custom Autoencoder Performance:* To evaluate the performance of the custom autoencoder model, a few random samples were collected from both the training dataset and a separate validation set. Figure 7 presents the visual results obtained from the autoencoder model on a dataset sample. The results demonstrate some of the intended behaviors, such as slight shifting and scaling of the image subject to resemble the simulated image more closely. However, the overall quality of the generated images is insufficient for practical use. While a larger model might yield improved results, the most notable aspect of this model's performance is its ability to successfully remove the background. Although this was not the expected outcome, it suggests that the autoencoder could be employed as a computationally efficient preprocessing step for background removal. Nevertheless, the model's outputs would

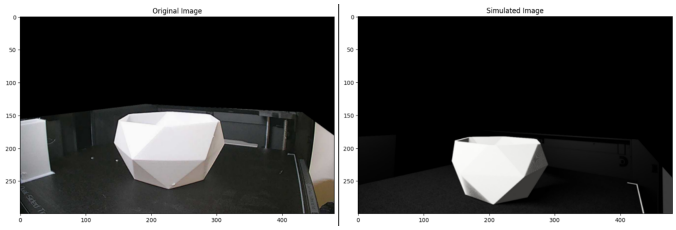


Fig. 8. Input captured image (left). Input rendered image (right).

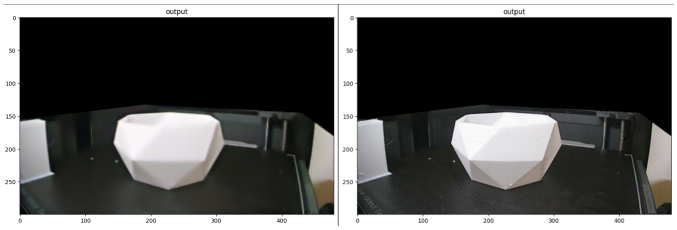


Fig. 9. VGG19 NST Result (left). Result with reversed inputs (right).

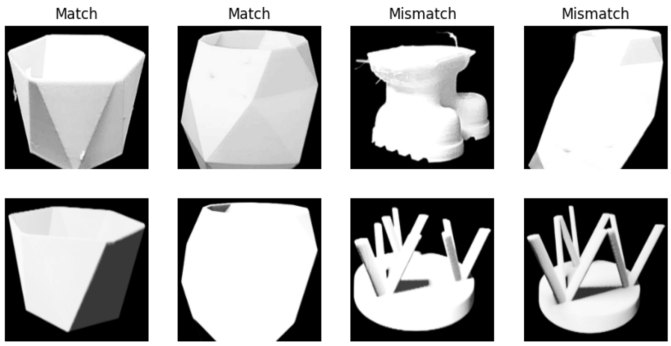


Fig. 10. Sample of training dataset with labels, captures, and renders.

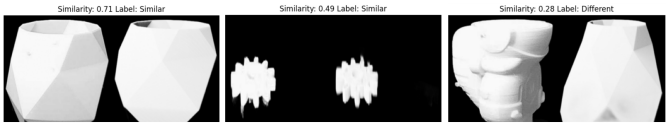


Fig. 11. Match result (left). Uncertain result (middle). Mismatch result (right).

not be suitable for change detection or similarity detection models in any meaningful capacity.

*2) Fine-Tuned VGG19 Performance:* An open-source VGG19 Neural Style Transfer (NST) runner script was utilized to deploy the fine-tuned model for image generation from the source and target images. Figure 8 displays the original images used as both the source and style images for the VGG19 NST operation. Figure 9 showcases the results of the VGG19 operations, using the capture as the source with the render as the style and vice versa, to determine which configuration would have a more significant impact on the result. The VGG19 model demonstrates more coherent style transfer, with clear indications of the style transfer effects, such as smoothing around the subject matter. Although the results are not entirely adequate for the proposed use case, adjusting model parameters, such as the percentage of the style image influencing the output and the per-image training epoch count, could potentially enhance the model's performance. However, due to the time-consuming and computationally intensive nature of running this model, these parameter optimizations were not carried out. Each image generation, using just 10 epochs, took 30 seconds on an RTX 3080ti GPU with CUDA acceleration. When deployed on IoT hardware, such a computational load would far exceed the acceptable runtime per image for real-time fault detection, assuming a model of this size could run at all.

### C. Fault Detection

The Siamese Neural Network (SNN) model developed for fault detection in this project was intended as a proof of concept, initially relying on the output from the Neural Style Transfer (NST) model for preprocessing. However, due to the suboptimal performance of the NST model, more rudimentary preprocessing methods were employed for the dataset. Despite this setback, the SNN, comprised of two Convolutional Neural Networks (CNNs) with a contrastive loss function, was able to train on the small dataset. Surprisingly, the dataset preprocessing yielded better results than anticipated, considering the challenges faced with the NST performance, which was ultimately not utilized. Figure 10 previews the dataset, showcasing correctly and incorrectly paired captures and renders, with appropriate labels for model training. The SNN model was trained using this dataset, and its performance was undoubtedly affected by the limited depth of the network and the small size of the dataset. However, despite these limitations, initial testing indicated that with further refinement, this approach has the potential to be effective for fault detection in additive manufacturing applications. Figure 11 shows results from this model where each comparison is given a similarity score ranging from 0 to 1. The results were plagued with examples such as the middle pair, where the images would be visually similar but the model would not be able to provide accurate results. Between this and a steady stream of false positives, this model in its current form is not good enough for this application. Although the current implementation of the SNN model leaves room for improvement, the results obtained from this proof of concept demonstrate the viability of using SNNs for fault detection in additive manufacturing. With further enhancements to the model architecture, dataset size, and preprocessing techniques, the performance of the fault detection system is expected to improve significantly.

## V. Conclusions

The exploration of Neural Style Transfer (NST) as a preprocessing tool for leveraging digital twins in additive manufacturing yielded mixed results. While the custom autoencoder failed to generate usable outputs, the fine-tuned VGG19 model showed promise, albeit with limitations. The style transfer from the VGG19 model functioned correctly, but the results were not sufficiently refined to validate NST as a standalone preprocessing solution for this digital twin project. However,

the autoencoder's ability to efficiently remove backgrounds from captured images presents a valuable finding, as it could serve as a computationally efficient preprocessing step for image normalization and transformation.

Despite the limitations of the current models, the tools and insights gained from this project lay the groundwork for future research in developing a robust preprocessing pipeline for change and similarity detection models in real-time additive manufacturing applications. The dataset generation process, a critical component of the project, performed exceptionally well, surpassing expectations. The successful creation of a comprehensive dataset of captured and rendered images provides a solid foundation for further exploration and refinement of the proposed approach.

To improve the usability of NST in this context, future work should focus on enhancing the image preprocessing pipeline by incorporating background removal and resizing techniques to isolate the subject matter and minimize extraneous elements that could hinder style transfer effectiveness. Additionally, optimizing the digital twin simulation script to render individual layers without backgrounds and stitching them together could significantly reduce dataset generation time from hours to minutes, making it more feasible for real-world deployment.

The Siamese Neural Network (SNN) fault detection results, although preliminary, show promise and could benefit from improved NST preprocessing. By refining the preprocessing techniques and leveraging the strengths of the dataset generation process, the overall performance of the fault detection system is expected to improve.

In conclusion, while the current models leave room for improvement, the concept of using NST and digital twins for real-time additive manufacturing fault detection has shown potential. The success of the dataset generation process, coupled with the insights gained from the preprocessing and fault detection experiments, provides a solid foundation for future research in this domain. By focusing on enhancing the dataset generation process and refining the preprocessing pipeline, this work paves the way for the development of accessible and efficient fault detection systems that can revolutionize the additive manufacturing industry.

## References


[1] N. Bharti, "3d printing in makerspaces: Health and safety concerns.," *Issues in Science and Technology Librarianship*, no. 87, Sep. 2017. DOI: 10.29173/istl1712. [Online]. Available: https://journals.library.ualberta.ca/istl/index.php/istl/article/view/1712.

[2] M. Baechle-Clayton, E. Loos, M. Taheri, and H. Taheri, "Failures and flaws in fused deposition modeling (fdm) additively manufactured polymers and composites," *Journal of Composites Science*, vol. 6, no. 7, 2022, ISSN: 2504-477X. DOI: 10.3390/jcs6070202. [Online]. Available: https://www.mdpi.com/2504-477X/6/7/202.

[3] A. Petsiuk, H. Singh, H. Dadhwal, and J. M. Pearce, "Synthetic-to-real composite semantic segmentation in additive manufacturing," *Journal of Manufacturing and Materials Processing*, vol. 8, no. 2, 2024, ISSN: 2504-4494. DOI: 10.3390/jmmp8020066. [Online]. Available: https://www.mdpi.com/2504-4494/8/2/66.

[4] M. Pantelidakis, K. Mykoniatis, J. Liu, and G. A. Harris, "A digital twin ecosystem for additive manufacturing using a real-time development platform," *The International Journal, Advanced Manufacturing Technology*, vol. 120, pp. 6547–6563, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246385150.

[5] J. An, T. Li, H. Huang, J. Ma, and J. Luo, "Is bigger always better? an empirical study on efficient architectures for style transfer and beyond," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023, pp. 4073–4083. DOI: 10.1109/WACV56688.2023.00407.

[6] W. Fan, J. Chen, J. Ma, J. Hou, and S. Yi, "Styleflow for content-fixed image to image translation," 2022. arXiv: 2207.01909 [cs.CV].

[7] R. Abdal, P. Zhu, N. J. Mitra, and P. Wonka, "Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows," *ACM Trans. Graph.*, vol. 40, no. 3, May 2021, ISSN: 0730-0301. DOI: 10.1145/3447648. [Online]. Available: https://doi.org/10.1145/3447648.

[8] G. R. Koch, "Siamese neural networks for one-shot image recognition," 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:13874643.

[9] Z. Alaverdyan, J. Jung, R. Bouet, and C. Lartizien, "Regularized siamese neural network for unsupervised outlier detection on brain multiparametric magnetic resonance imaging: Application to epilepsy lesion screening," *Medical Image Analysis*, vol. 60, p. 101 618, 2020, ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2019.101618. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841519301562.

[10] C. Pottier, J. Petzing, F. Eghtedari, N. Lohse, and P. Kinnell, "Developing digital twins of multi-camera metrology systems in blender," *Measurement Science and Technology*, vol. 34, no. 7, p. 075 001, Mar. 2023. DOI: 10.1088/1361-6501/acc59e. [Online]. Available: https://dx.doi.org/10.1088/1361-6501/acc59e.

[11] T. Chiu and D. Gurari, "Photowct2: Compact autoencoder for photorealistic style transfer resulting from blockwise training and skip connections of high-frequency residuals," in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2022, pp. 2978–2987. DOI: 10.1109/WACV51458.2022.00303. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/WACV51458.2022.00303.

[12] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2414–2423. DOI: 10.1109/CVPR.2016.265.

[13] S. Bangar, *Vgg-net architecture explained*, Jun. 2022.

[14] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 1735–1742. DOI: 10.1109/CVPR.2006.100.

[15] T. Sonnenberg, A. Stevens, A. Dayerizadeh, and S. Lukic, "Combined foreign object detection and live object protection in wireless power transfer systems via real-time thermal camera analysis," in *2019 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2019, pp. 1547–1552. DOI: 10.1109/APEC.2019.8721804.