

NOVA

IMS

Information
Management
School

MDSAA

Master Degree Program in
Data Science and Advanced Analytics

Building a Fused Deposition Modelling (FDM) 3D Printing Visual Defect Detection System, Part 1: Creation of a Dynamic Imaging System, a Pixel-wise Segmentation Dataset with a Hybrid Synthetic Data Creation Method, and a Semantic Segmentation Algorithm with the SegFormer Deep Learning Model

Doyun Shin

Project Work

presented as partial requirement for obtaining the Master Degree Program in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

**BUILDING A FUSED DEPOSITION MODELLING (FDM) 3D PRINTING
VISUAL DEFECT DETECTION SYSTEM, PART 1: CREATION OF A
DYNAMIC IMAGING SYSTEM, A PIXEL-WISE SEGMENTATION
DATASET WITH A HYBRID SYNTHETIC DATA CREATION METHOD,
AND A SEMANTIC SEGMENTATION ALGORITHM WITH THE
SEGFORMER DEEP LEARNING MODEL**

by

Doyun Shin

Project Work report presented as a partial requirement for obtaining the Master degree in Advanced Analytics, with a Specialization in Business Analytics

Supervisor / Co Supervisor: Mauro Castelli, PhD.

Co-Supervisor: N/A

ACKNOWLEDGEMENTS

This project was conducted in cooperation with Version3D (*eenmanszaak*, sole proprietorship) based in Eindhoven, the Netherlands. The company's founder, Rob van den Nieuwelaar, has supported the development of the hardware design of this project with a provision of a 3D printer, mechanical insights, and CAD drawing assistance. He also shared his knowledge in the 3D printing community, which has helped setting the scope of this project and deciding on many UI related decisions.

ABSTRACT

As a part of an effort to develop a surface defect detection system for FDM 3D printed objects, this work project studies the application of the SegFormer network to semantically segment 3D printed objects. The project also showcases an affordable and accessible imaging system designed for the surface defect detection system, to support the decisions made during the segmentation task and to be used to evaluate the segmentation models. To achieve this, the first-ever pixel-wise annotation dataset of 3D-printed object images was created. Model-O1, a SegFormer MiT-B0 model trained on this dataset with minimal data augmentation resulted in an Intersection-over-Union score of 87.04%. A synthetic data creation method that caters to the nature of 3D printed objects was also proposed, which expands upon existing synthetic data creation methods. The model trained on this dataset, Model-A2, achieved an IoU score of 89.31%, the best performance achieved among the models developed in this project. During the evaluation of the model based on the inference results, Model-A2 was also identified to be the most practical model for building a surface defect detection system.

KEYWORDS

3D printing; Semantic segmentation; Vision Transformer; 3D print dataset

INDEX

1. Introduction	1
2. Literature Review	4
3. Methodology.....	8
3.1. Imaging System.....	8
3.1.1. Components.....	8
3.1.2. Functionality	8
3.2. Semantic Segmentation.....	11
3.2.1. Dataset	11
3.2.2. Network Selection.....	18
3.2.3. Optimizer and Learning Rate	18
3.2.4. Loss Function	19
3.2.5. Evaluation Metrics	20
3.2.6. Training	21
4. Results and Discussion	26
5. Conclusion	33
6. Limitations and Recommendations for Future Works.....	34
7. References.....	37
8. Appendix A. Inference results on less visible 3DP features	42
9. Appendix B. Inference result on severely distorted features	43
10. Appendix C. Inference results in the imaging system environment.....	44
11. Appendix D. Model-B1 convergence and validation IoU over 13 epochs	45

LIST OF FIGURES

Figure 1.1 – Critical Failures	1
Figure 1.2 – Low Visibility Defects.....	2
Figure 2.1 – Slicer Layer Pattern	5
Figure 2.2 – Examples from the FLAME Dataset Used in Ghali et al. (2022)	6
Figure 3.1 – Imaging System Camera Positions	8
Figure 3.2 – Imaging System Prototype	9
Figure 3.3 – Images Taken with the Imaging System	10
Figure 3.4 – Ambiguous Defective Area and Multiple Defects in a Surface	12
Figure 3.5 – Mini-Batch Loss Fluctuation over Increasing Learning Rate	19
Figure 3.6 – Confusion Matrix	20
Figure 3.7 – Model-A Convergence and Validation IoU over 58 Epochs	23
Figure 3.8 – Model-G Convergence and Validation IoU over 42 Epochs	24
Figure 3.9 – Model-O Convergence and Validation IoU over 70 Epochs	24
Figure 4.1 – Inference Results on Multi-Colour Filaments.....	28
Figure 4.2 – A Sample of Model-A2 Inference Results.....	30
Figure 4.3 – Inference Results in the Imaging System Environment	31
Figure 6.1 – Complex Shape Annotation and Unexpected CVAT Behaviours.....	35

LIST OF TABLES

Table 3.1 – Composition of the Mixed Dataset.....	17
Table 4.1 – Top 3 models for each experiment based on the average IoU score.	26

LIST OF ABBREVIATIONS AND ACRONYMS

3DP object Throughout the study, an unofficial abbreviation “3DP object” will be used to refer to an object printed from a 3D printer.

CNN Convolutional Neural Network

FCN Fully Convolutional Network

FDM Fused Deposition Modelling

SDDS An unofficial acronym for Surface Defect Detection System used in this paper

1. INTRODUCTION

Version3D is a partner of Boston Scientific catering to the biomedical engineering firm's prototyping needs. With the expansion of the business, Version3D has invested in a belt-3D printer, which continuously extrudes filaments directly to a moving conveyor belt, without requiring the printed products to be manually removed after each print. Nonetheless, the new printer still requires constant human monitoring for failure and defect detection. Version3D expressed its need for an affordable surface defect detection system (SDDS) for 3DP objects for its belt 3D printer. Furthermore, Version3D has shared an insight that this is a common problem within the 3D printing community, as it consists of mostly individual hobbyists and small businesses that lack the capital to invest in the existing high-precision defect detection equipment or dedicated manpower. As such, this project was conducted to develop the foundation of SDDS for 3D printed (3DP) objects. More specifically, it aims to utilize SegFormer to semantically segment 3DP objects.

Before building an SDDS, a series of informal experiments were conducted to grasp the boundaries of possible achievements with available tools. During the process, it was found that critical defects such as spaghetti,¹ stringing, and severe under/over-extrusion could be detected through an object detector network YOLOv7 (C. Y. Wang et al., 2022) without significant commitment, trained on a small dataset of around 100 annotations per class. Figure 1.1 provides examples of these defects.

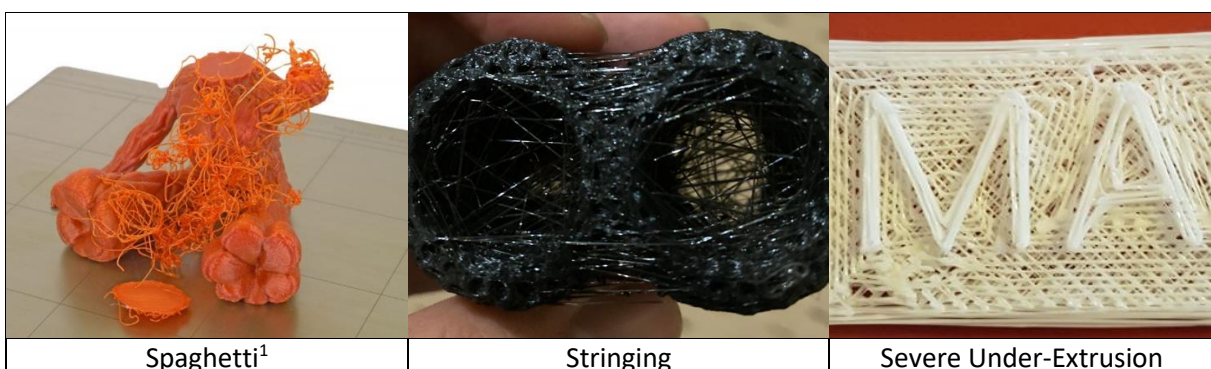


Figure 1.1 – Critical Failures

¹ This paper refers to defects by the most commonly used terms in various 3D printing community. There is no single source that compiles all the existing defect types using unified terms, due to the complexity and diversity of 3D print defects. Some attempts were made though, such as Simplify3D (2019), which was considered for the terms used in this paper.

However, small features such as zits, holes, and minorly inconsistent extrusion could not be detected reliably, simply because their visibility was highly susceptible to changes in camera focus, angle, and lighting. Some of these defects are illustrated in Figure 1.2. Based on this finding, the author proposes

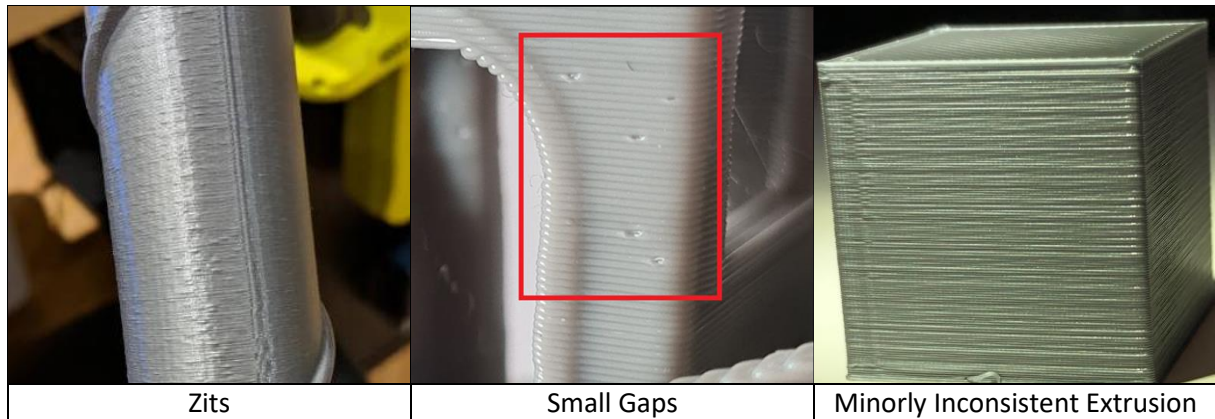


Figure 1.2 – Low Visibility Defects

a 2-stage defect detection system. The first stage would utilize a low-quality camera to detect critical failures in real-time to halt the printer operation if needed. Such a failure management system is already available, but only for spaghetti defects (Obico., (2019); Mech Solution Ltd., (2022)).² The second stage would conduct a post-print inspection with a higher resolution camera with focus control. At this stage, multiple images of the 3DP object will be taken in varying camera focus values. Then, the less apparent defects can be detected in these images. In some images, defects will be found, and, in some images, they will not be. But with a successful semantic segmentation, higher weight can be given to the defects identified in the region with higher sharpness measured by the Laplacian method. This process is only necessary for building an *affordable* imaging system—high-cost cameras are usually equipped with a wide range of aperture control which allows a wide depth of field, leading to high sharpness all around the focused object even from close distance from the object. Ultimately, this project experiments with semantically segmenting 3DP objects with Xie et al.’s (2021a) SegFormer network. Although the core aspect of this paper is the segmentation task, a functioning imaging system

² During the development of this project in August 2022, 3DQue showcased QuinlyVision, a failure management system capable of detecting many more types of defects. Nonetheless, the performance still remains largely problematic for small defects (3DQue, 2022). Hence, the motivation and the relevance of this project remain unhindered.

was also developed for the SDDS, and the paper illustrates the hardware aspects of the system to a limited extent as many of the decisions taken for the segmentation study are based on the hardware design. Moreover, to achieve the segmentation task, this project created a pixel-wise segmentation dataset which consists of 3,668 FDM 3DP object images. The SegFormer model trained on this dataset with a minimal augmentation resulted in an Intersection-over-Union (IoU) score of 87.04%, making the dataset to be, in this author's knowledge, the first-ever 3DP object dataset large enough to train a functional semantic segmentation model. To achieve a better performance, the data was augmented via synthetic data creation. During the process, the author proposes a hybrid method utilizing the existing synthetic data creation methods, while combining it with further data augmentation transformation techniques. Using the resulting training dataset, Model-A2, the best model developed by this project, has achieved an IoU score of 89.31%.

2. LITERATURE REVIEW

To the author's knowledge, there has not been any study conducted on the semantic segmentation of 3DP objects. Hence, the author has investigated segmentation studies in general with a special emphasis on smoke segmentation studies. Smoke is a distinctive subject in that it has a non-constant shape and colour. 3DP object adds an extra complication to the study in that its shape and colour are even more fluid than smoke and fire. Even the textures can be drastically different from one another depending on the printing pattern, lighting, image focus, and presence and severity of defects.

In the field of semantic segmentation, convolutional neural networks (CNN) have rendered many of the traditional methods obsolete (Guo et al., 2017), such as random forest and conditional random field (CRF) approaches, with a significant performance improvement (Mo et al., 2022). The publication of AlexNet (Krizhevsky et al., 2012) was quickly followed by the study of Fully Convolutional Network (FCN) by Long et al. (2015) which became the foundation of many models to follow. The development of U-Net (Ronneberger et al., 2015), which has become the benchmark of the medical image segmentation task, has inspired other U-shape networks with its simple structure and strong generalization (Wang et al., 2022). Many more state-of-the-art models, such as VGG (Simonyan & Zisserman, 2014) and Resnet (He et al., 2016) are based on CNN. The inherent weakness of CNN, though, quickly became the limiting factor to the progress. By its design, long-range dependencies require large receptive fields. Some efforts were made to tackle this issue, such as atrous convolution layers implemented in Deeplab by Chen et al. (2018), non-local operations implemented by Wang et al. (2018), and the attention gate model by Schlemper et al. (2019).

Another approach to deal with the intrinsic locality of convolution operations was using Transformers, an alternative architecture to CNN, to exploit their global self-attention mechanisms (Chen et al., 2021). This Transformer approach for computer vision is rather new, transpired by Dosovitskiy et al.'s (2020) Vision Transformer (ViT) only in 2020. Since then, many studies have experimented with Transformer with a special focus on its self-attention mechanism, TransUNet (Chen

et al., 2021) being one of the first segmentation models. Self-attention modules are a type of attention mechanism which aggregates the information of the entire input sequence and update each component of a sequence. With this mechanism, long-range dependencies between sequence elements can be captured. As a result, it avoids the need for large receptive fields which incurs a substantial computational cost. In addition, albeit there are some disagreements such as Bai et al., 2021, currently the dominant opinion is that Transformer is more robust to various corruptions and is better at uncertainty estimation than CNNs, and self-attention is understood to be the probable cause of these strengths (Pinto et al., 2022; Zhou et al., 2022; Mahmood et al., 2021). As such, studies such as Wang et al. (2022) are heading in a new direction in the field of image segmentation and classification by attempting to exploit both Transformer’s edge in capturing global information with CNN’s strength in learning local information.

Another noteworthy Transformer-based segmentation model is SegFormer by Xie et al. (2021a). The study describes this model to be “a simple, efficient, yet powerful semantic segmentation framework.” The model combines a hierarchical Transformer model with a lightweight multiplayer perceptron (MLP) decoder. The most remarkable point of SegFormer is in the hierarchical Transformer encoder design, which produces multiscale features, which scraps the need for positional encoding which can decrease the model’s performance when inferencing on images with resolutions different from the training images. Furthermore, it implements an all-MLP decoder instead of other complex decoders. The MLP decoder performs as an information aggregator from multiple layers, which in turn integrates local and global attention at the same time.

The nature of 3DP objects, in theory, aligns well with the strength of SegFormer. 3DP objects, with their non-constant shape, colour, and texture, can be identified only by their slicer layer pattern shown in Figure 2.1. This pattern refers to the pattern formed naturally during the

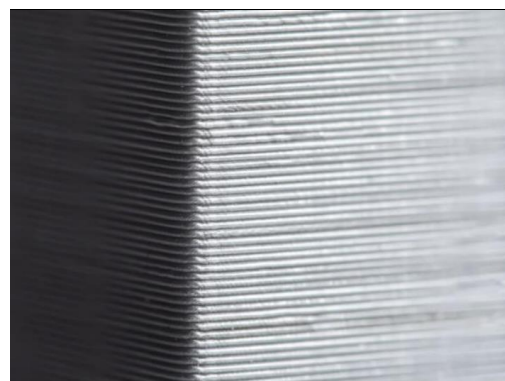


Figure 2.1 – Slicer Layer Pattern

FDM printing process. However, depending on the lighting, focus, and image quality, these patterns are often visible only in parts of the object. If a model fails to learn from the global information and correlated these pattern-less areas of the object with 3DP objects only based on the local information, it is highly unlikely the model will successfully differentiate 3DP objects from general plastic objects. With Transformer’s ability to capture global information, the model may correlate the presence of 3D printing patterns (such as the slicer layer pattern) with other pixels that do not form these patterns but otherwise share similar pixel values and identify these pixels to be part of the 3DP object. In fact, Transformer has already been implemented for smoke and fire segmentation with promising results. For instance, Ghali et al.’s (2022) experiment with TransUNet-R50-ViT scored an F1-score of 99.90% by extracting local and global features using a hybrid backbone consisting of CNN, R-50, and pre-trained ViT Transformer. This result outperforms the score of CNN models they experimented with, such as U-Net (F1 = 99.00%) and EfficientSeg which uses MobileNetV3 blocks on a U-Net structure (F1 = 99.66%). The study also states that their Transformer model outperformed the CNN models with higher accuracy, however, considering aerial images of smoke and fire tend to be highly class-imbalanced as shown in Figure 2.2, accuracy is unlikely to be an appropriate evaluation metric to compare the models.



Figure 2.2 – Examples from the FLAME Dataset Used in Ghali et al. (2022)

In another study, Zheng et al. (2022) present three models using a Transformer encoder. Their best model, Smoke-U-Net which adopts both Multi-Scale Residual Group Attention (MRGA) and Global

Features Module (GFM) subsequently outperformed multiple CNN models assessed, namely, SegNet, PSPNet, U-Net, and DeepLab v3+. The study reports DeepLab v3+ has the highest mIoU score among these CNN models in all scales (large 91.03%; medium 88.61%; and small 85.25%). Smoke-U-Net however, outperforms DeepLab v3+ on all three scales, by 3.94%, 3.54%, and 3.11%, respectively. These successes further support the use of a Transformer-based model to semantically segment 3DP objects.

3. METHODOLOGY

3.1. IMAGING SYSTEM

The main scope of the project is the semantic segmentation aspect. However, since many of the decisions made during the segmentation study were motivated by the hardware design of the imaging system, this section briefly covers the specifications of the imaging system. Simplicity, accessibility, and affordability were the main concerns in building the system.

3.1.1. Components

The structure was entirely 3D printed on an FDM printer, which allows any potential user to print their own system. Other requirements are a stepper motor, LED strips (5000K to 6000K colour temperature for white balance), nuts and bolts, an Arduino, white baking sheets as a light diffuser, a white paper as the backdrop, a power adapter, and a 1080p webcam with focus control. For this project, Logitech c920 was used. This webcam used to be considered affordable, often priced under USD 50 (PCGamer, 2016). But during the COVID-19 period, webcam demands have surged, increasing the price significantly to EUR 70 at the time of this project's writing. Due to budget reasons, only one cheaper camera was tested, eMeet Nova webcam priced under EUR 30. It was compatible with our system, although different camera configuration values had to be used when controlling it through the OpenCV library. This paper solely focuses on the Logitech c920, as it was not possible to find some necessary specification information for eMeet webcam. Old smartphones were briefly considered due to their powerful cameras, but they were disregarded due to weight and size issues.

3.1.2. Functionality

The system currently recommends to capture images in 5 positions. The camera, at its idle position

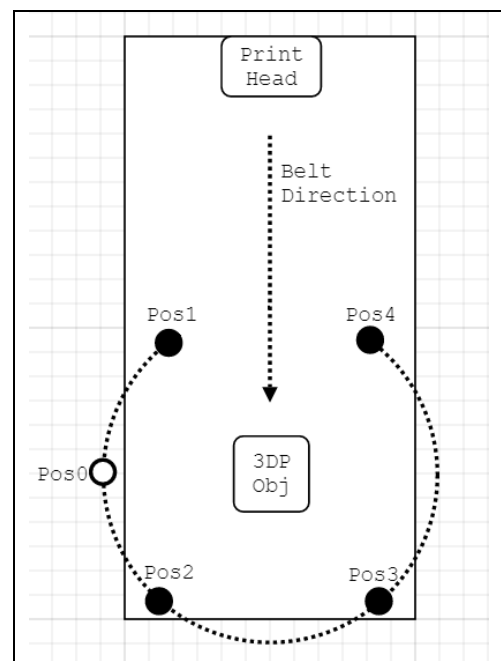


Figure 3.1 – Imaging System Camera Positions

(Position 0), faces the conveyer belt at a perpendicular angle and only a real-time object detection network, such as YOLOv7, would be inferencing. When a 3DP object is detected and remains stationary around the centre of the display, the motor rotates the camera to position 1 to capture photos at different focuses then moves onto Positions 2, 3, and 4 to repeat the imaging process, and then returns to position 0 as illustrated in Figure 3.1. Position 1 through Position 4 span across 270 degrees, to prevent the camera from a potential collision with the next incoming object, which may happen if the printer parameter is set incorrectly. The test was done with a simple colour segmentation method to detect objects, using only blue and green filaments for simplicity. The system did not show any failure during a 24-hour continuous printing test.

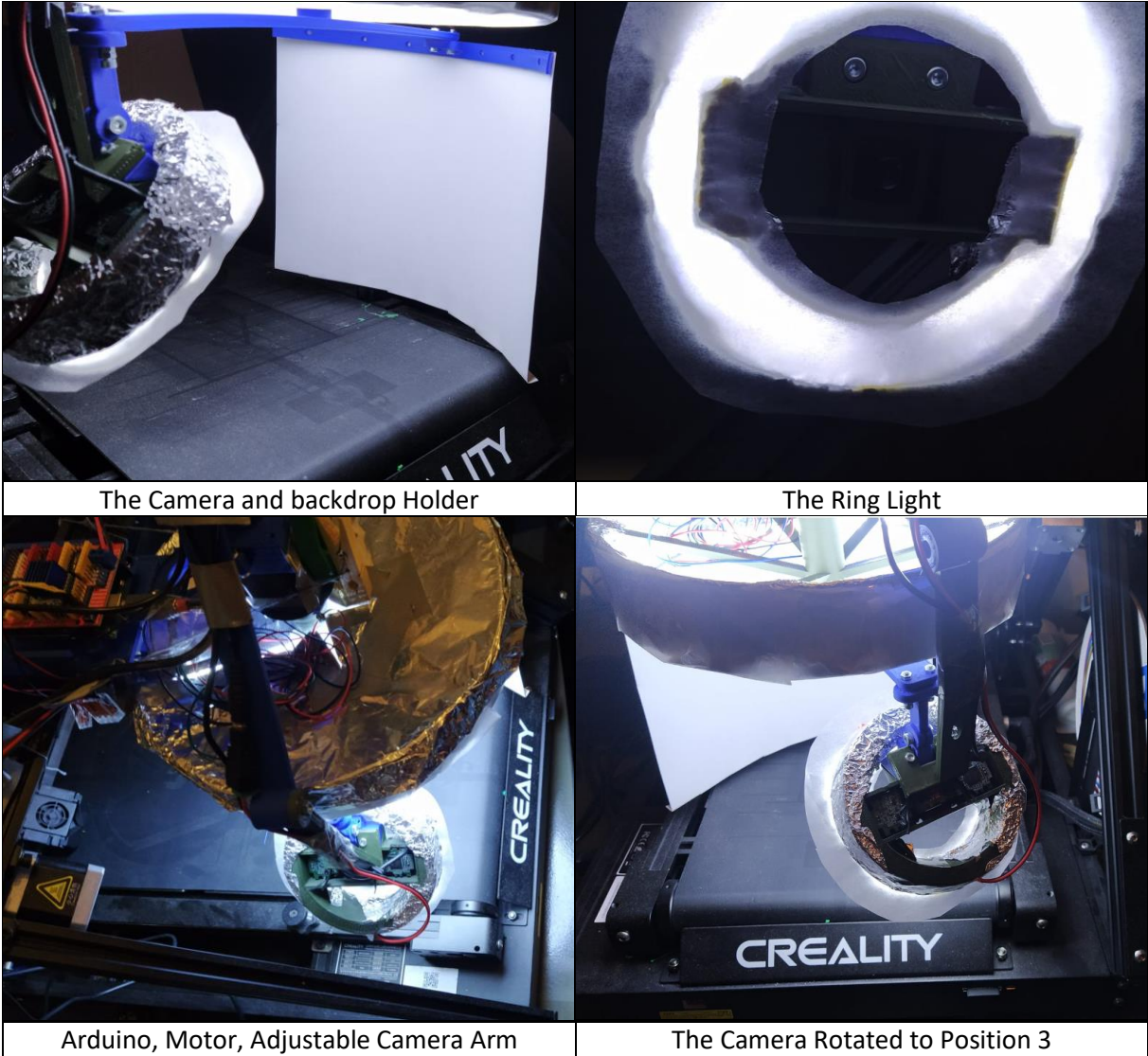


Figure 3.2 – Imaging System Prototype

The camera and the backdrop are connected by a bar, which means the backdrop always stays on the opposite side of the camera. The light diffuser at the ceiling light prevents this pole from casting a shadow. Together with the ring light from the camera and reflected light off the backdrop, they minimize as much of the 3DP object's shadow as possible while illuminating the object's details clearly. Figure 3.2 shows the prototype of the imaging system and Figure 3.3 shows some of the test images taken by the system. The purpose of the camera arm in Figure 3.2 is to better distribute the weight while allowing manual adjustment of the vertical camera angle.



Figure 3.3 – Images Taken with the Imaging System

An artificial frame rate control was also implemented into the camera interaction. To the author's knowledge, there is no commercially available webcam with adjustable frame rate control. This is an issue since image processing time often exceeds the interval between incoming frames, which leads to new images piling up while the previous image is still being processed. This gap only increases over time. As such, this project can exploit the fact that defects are formed slowly during the 3D printing process, which means real-time monitoring is not required. Therefore, a multiprocessing approach was employed where a thread is reserved for only capturing incoming frames. The main process, then, purges all the frames collected so far in the frame-capturing thread every fixed interval (for example, 2 seconds) and grabs the next incoming frame and processes that image, then repeats the process. This frame rate control, in turn, eliminates the lag between the image being currently processed and the image being currently captured. Our test has shown that the system is robust to varying brightness of the surrounding light. Unless the sunlight is directly hitting the system, the images

were quite consistent with the brightness and white balance. This robustness was achieved by carefully selected light temperature, and low camera exposure on high LED light intensity.

3.2. SEMANTIC SEGMENTATION

3.2.1. Dataset

A major limitation to conduct this project was the absence of pixel-wise 3DP object annotation data. As the dataset had to be built from the ground up, a significant portion of the project time was spent on this effort.

3.2.1.1. Data Collection and Preparation

A majority of the 3DP object images were obtained via web scraping, and a small number of images taken from the imaging system were added to the dataset. Initially, 27,268 images were collected based on 3D print defect keywords, such as “over extrusion” and “layer split.” This approach was inspired by two reasons. Firstly, a simple search of “3D printed object” often led to images of other related topics rather than 3D printed objects themselves. Secondly, individuals who uploaded 3DP object images with defects in the title often took a close-up image of the object to clearly show the defects. Such distance to the camera closely resembled the camera proximity of the developed imaging system.

Following the data collection, duplicate images were removed through *Duplicate Cleaner*, a software which can detect exact duplicates as well as different-size duplicates. Then, the images were manually reviewed and *unusable* images were discarded subjectively. These images include non-3DP objects, non-FDM-based 3DP objects, transparent filament, single-layer prints, 3DP objects too close or too far from the camera, and heavily obstructed 3DP objects. In addition, it is important the trained network does not confuse a 3DP object with general plastic objects. Therefore, 3DP objects that are too blurry or images with a very low resolution where very little print detail (such as “slicer layer patterns” which was presented in Figure 2.1) was visible were also excluded. Furthermore, severe spaghetti defects were excluded from the dataset, as they are vastly different from other 3DP objects.

These severe spaghetti defects would be detected independently from 3DP object segmentation, as discussed in the introduction.

After the filtering, 8470 images remained, and they were organized based on the type of defects most perceptible in the image. A significant bias may have been induced into the dataset at this step, as there were often multiple types of defects present in the same 3DP object. The defects were also built on top of each other, creating defects that are difficult to categorize. Furthermore, the severity and the area of defects are not definitive. Figure 3.4 illustrates such cases.

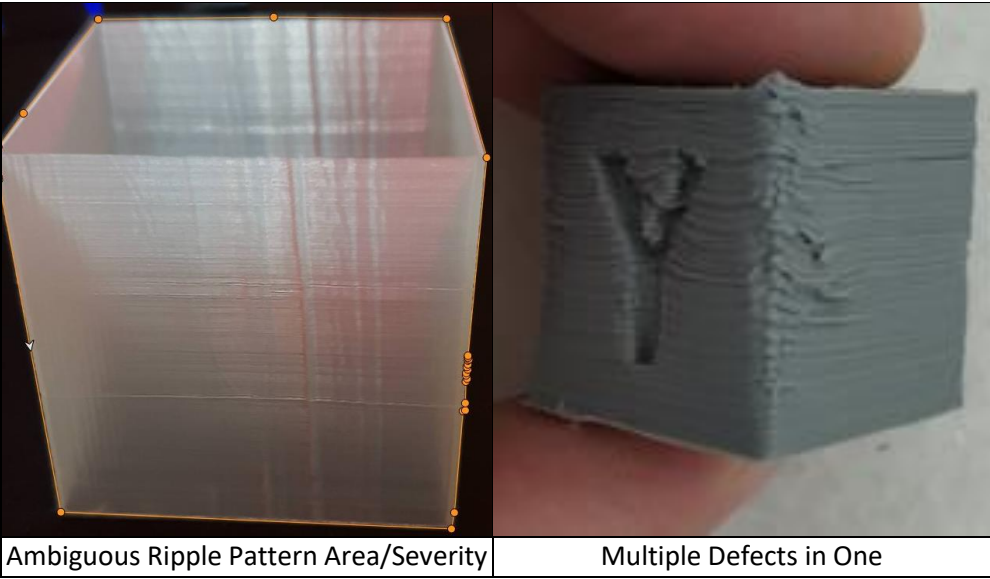


Figure 3.4 – Ambiguous Defective Area and Multiple Defects in a Surface

It would be ideal to precisely identify the severity, type, and area of defects to improve the class balance to avoid the model from associating specific types of defects with the characteristics of 3DP objects. However, the scope of this project is segmenting 3DP objects, not differentiating types of defects. Additionally, despite the defects, the pattern present in all 3DP objects (except for severe spaghetti defects) when imaged in an appropriate setting is the slicer layer pattern, which could potentially counteract the imbalance of defect types present in the dataset when segmenting the 3DP objects.

A more significant bias in the dataset was the absence of uncommon filaments, such as transparent or patterned filaments, and 3DP objects printed with multiple different colours of

filaments. Transparent 3DP objects were excluded entirely, as the slicer layer pattern is also hardly visible in this type of filament, while they are extremely uncommon (there was a total of 7 images found in the initial 27,268 images collected). The intuition was that it would induce an extra complication to the already non-constant nature of 3DP objects. Patterned and multi-coloured 3DP objects were also rare but as the slicer layer pattern was usually visible, they were included in the dataset. Nevertheless, a performance issue with segmenting these two types of 3DP objects was expected due to the rarity of these cases.

Finally, 3900 images were annotated with polygons through Computer Vision Annotation Tool (CVAT) and the masks were stored in PNG format. Initially, the dataset was split into 8:2 ratios for training and validation purposes. However, during the later stage of the project, some images with corrupt Exchangeable Image File Format (EXIF) data were discovered. Furthermore, images with an EXIF rotation flag caused confusion throughout the study, as CVAT, Windows Photos, python Pillow library and python OpenCV library all handle it differently. After discarding these images, the actual data used for this project consisted of 3093 training images and 575 validation images. The mask images were exported in object segmentation format (as opposed to class segmentation format), which can be used for instance segmentation projects. This study converted the mask images to (binary) class segmentation for the semantic segmentation task, with pixel value 1 representing 3DP prints and 0 representing the background.

3.2.1.2. Training Data Augmentation via Synthetic Data Creation

For this study, a synthetic data creation approach was used to augment the data. The approach was inspired by Yuan et al. (2019) and utilizes noise object assets from P, A. (2022). Nonetheless, this project's synthetic data creation method differs significantly from the two. Yuan et al. (2019) placed computer-generated subject images (i.e., smoke) at the centre of a randomly selected image from the Places dataset, a large dataset of background images compiled by Zhou et al. (2017). On the other hand, P, A., 2022 utilizes only 30 different background images but created diverse composite

background images by randomly placing 107 different noise object images onto the background images. Then, the subject images (i.e., battery, lightbulb, and padlock) were randomly placed on those composite background images.

This project combines the two methods and incorporates the image augmentation process concurrently. The dataset is a combination of the below three types (type 1, 3, 4) of data. For all size reduction, OpenCV's INTER_AREA interpolation method was used and for all enlarging, INTER_CUBIC was used, following the recommendation provided in the OpenCV documentation (OpenCV, 2022). In any case, the final resized dimensions do not differ from the original dimensions by more than 15%. This decision was made, as the slicer layer pattern is an important signal to identify 3DP objects. For non-90 degrees rotations, different methods were tested but the difference was hardly noticeable by human eyes for this dataset. Ultimately, INTER_CUBIC was chosen based on Lehmann et al. (1999).

Type 1 Data:

Type 1 data is the original images. All images with a height exceeding 1024 pixels were resized to 1024 pixels while keeping the height-width ratio. There was no image with a width exceeding 2048 pixels after the resizing. Then, any image with a width exceeding 1024 pixels was padded to 2048x1024 dimensions and sliced into 1024x1024 dimensions. Any image with a width shorter than or equal to 1024 was padded to 1024x1024 dimensions.

Type 2 Data (discarded):

Type 2 data adds an augmentation element via the Albumentation library. Random flip, 90-degree rotation, and minor brightness (up to 0.1 scales), contrast (0.15), saturation (0.15), and hue (0.1) modifications to each slice of Type 1 data. Brightness was changed only conservatively since it was identified brightness often severely damaged the visibility of print patterns when altered too much. Type 2 data, however, was later discarded due to the VRAM limitation and training speed.

Synthetic Data (Type 3 and Type 4):

Type 3 and Type 4 data were synthetically created, as the author hypothesized the model will generalize better if trained on a more diverse range of background compositions. If the project aims to build an SDDS only for Version3D's 3D printer, the best option would be to take images of the belt printer environment and train the model with those images. However, the project aims to be scalable for future usage in different environments, hence various background images were used. A total of 377 background images were collected, consisting of google image search results, YouTube video screenshots, photos of the 3D printer environment taken with this project's imaging system, and some indoor and outdoor photos personally taken by the author. A particular emphasis was given to obtaining images of different materials, such as metal, glasses, plastic, leather, rubber, and cloth. Other search terms include workbench, tools, 3D printer, print bed, and conveyer belt.

These background images were resized and/or sliced to form 4242 background images of 1024x1024 dimensions and 1153 images of 2048x1024 dimensions. Then, up to 10 random noise objects were randomly placed with varying degrees of overlap onto a randomly selected 1024x1024 background image. For the 2048x1024 images, the max number of noise objects was set to 15. Although the dimensions were double the 1024x1024, when the max number of noise objects was set to 20, the composite background images often appeared to be highly cluttered with noise objects. 104 images of different noise objects were utilized. During the composition process, each noise object was transformed independently and individually with Albumentation Rotate, Flip, ColorJitter (hue, contrast, brightness, saturation, channel shift), Blur, and Resize. The background images were also modified similarly, except the rotation method was RandomRotate90. The ceiling values for the colour changes and the resize limit were set generously to create diverse composite background images. 10,000 composite background images were made for 2048x1024 and 1024x1024 dimensions each.

Finally, up to 5 random 3DP object images were cropped out and placed onto the 2048x1024 composite background images. A random transformation composition of Rotate, Flip, ColorJitter, and ChannelShuffle was applied to each cropped-out image before the placement. The ColorJitter values

here are identical to the Type 2 Data above. Also, random resizing was done while maintaining the height-width ratio and the resized dimensions do not differ more than 15% from the original dimensions. Afterwards, varying degrees of GaussianNoise were applied to the whole image. The probability of GaussianNoise modification was set to 20% since the original images already contained some noises. Furthermore, all the images were saved as JPEG to purposefully induce JPEG compression loss. The intuition was that Logitech C920, like most other webcams, uses the h.264 encoding method, a lossy compression. Although JPEG and h.264 compression differ from each other, it was a simple way to cause some compression loss to the training dataset. The final images were then sliced into two 1024x1024 patches, forming Type 3 Data. This process was repeated several times to generate multiple sets of data with varying Albumentation modifications.

Type 4 Data is very similar to Type 3. However, it was identified in Type 3 Data, sometimes a 3DP object would be sliced in a way that the 1024x1024 patch contained only a small, unrecognizable part of the object. The biggest issue was when that sliced part happens to be a blurry part of the object, without any visible slicer layer patterns. This could cause confusion in differentiating 3DP objects and general plastic objects during the model training. While these images were not removed from the dataset, Type 4 Data was introduced to counter the balance. In Type 4 Data, cropped-out images of 3DP objects are not randomly placed onto composite background images. Instead, the height and width of the bounding box were measured, and if the image could fit inside a 1024x1024 composite background image by reducing the size up to 15%, it was reduced and placed at the centre of the composite background image. Otherwise, it was placed at the centre of a 2048x1024 composite background image, then sliced into 1024x1024 patches, followed by a RandomRotate90 transformation. Other Albumentation transformations were identical to Type 3 Data. This process was also repeated several times.

Initially, more than 200,000 images were created. However, due to hardware limitations, only 20,300 images were used for training SegFormer. Type 2 was discarded completely, as such

transformations can be done with the data loader in the Pytorch library every epoch. Table 3.1 summarizes the composition of the final dataset used for the model training. This dataset will be referred to as “Mixed Dataset” throughout the paper.

Data Type	Count of Slices (1024x1024)
Type 1	4,686
Type 2	0
Type 3	6,260
Type 4	9,354
Total	20,300

Table 3.1 – Composition of the Mixed Dataset

3.2.1.3. Validation Data

The validation images were resized only based on the height, to be within 1080 pixels, while keeping the width-height ratio. Afterwards, if the image’s height was higher than 1024, the image was cropped equally from the top and the bottom. Then, they were padded to have 1024x1024 or 2048x1024 dimensions depending on their original dimensions, to be sliced into 1024x1024 patches. This approach was adopted as the final imaging system will be using a 1920x1080 resolution camera. To align with the SegFormer model to be trained on 1024x1024 images, the 1920x1080 will be sliced into three overlapping patches for inferencing, with 50% overlaps. 50% overlap was chosen, as that is the planned overlap size for the imaging system’s sliding window inference method. Images with a width exceeding 1024 could have been sliced with an overlap without being padded to 2048. However, that would cause an issue for images with a width that exceeds 1024 pixels only by a small margin, as the slices would be almost identical. The image set consisting of was saved in PNG format. All images containing less than 250 pixels of 3DP object area were removed from both the train and validation datasets.

3.2.2. Network Selection

After a literature review, SegFormer was selected. The intuition was that, since 3DP objects are often indistinguishable from other general plastic objects when the object details are unclear, it is important for the model to associate the presence of slicer layer patterns with surrounding pixels. If the surrounding pixels do not form slicer layer patterns, if they share similar traits with the slicer layer pattern region, such as the colour, they are likely to be a part of the 3DP object.

3.2.3. Optimizer and Learning Rate

The learning rate is arguably the most important hyperparameter in deep learning (Bengio et al., 2017, p. 376). However, in developing the SegFormer, Xie et al. (2021a) do not place great emphasis on identifying the optimal learning rate. Instead, a polynomial learning rate decay schedule is used on the initial value of 0.00006 with factor 1.0, with AdamW optimizer. Such an approach, where the initial learning rate is set very low and even further decreased with learning rate decay, was possible for Xie et al. (2021a) because they trained for a large number of epochs, 160K iterations for Cityscapes (p. 6). For this project's case, however, a more optimal initial learning rate had to be identified as the available computing power limited the number of epochs to be trained. As a reference, it took around 1.5 hour per epoch for training and 0.5 hour per epoch for validation.

Same as the original SegFormer, AdamW optimizer and polynomial learning rate schedule were adopted for this project. However, the learning rate was set at 0.001, partially motivated by Smith, L. N.'s (2017) approach. In section 3.3 of his work, he discusses a method to find "reasonable minimum and maximum boundary values" for cyclical learning rates. His method consists of training the model multiple epochs while linearly increasing the learning rate while measuring the accuracy against the learning rates. The minimum boundary value would be the learning rate that is approaching the peak accuracy, and the maximum value would be the learning rate when the accuracy starts falling. However, AdamW applies local adaptive learning rates, whereas his method was proposed for a cyclical learning rate (CLR) policy with a global learning rate. Nevertheless, despite their fundamental

differences, he suggests a combination of CLR and Adam is still possible, and experiments with the minimum and maximum values found from the above test in section 4.4 of his work. Hence, this project considers his method but utilizes Fast.ai’s (2016) adaptation version of it, where the learning rate is exponentially increased, and the loss is measured for each mini-batch over a single epoch. The fastai library was not directly applied, but the source code was referenced to be utilized for this study’s models. The first run was done starting from the learning rate of 10^{-6} increasing to 1, first with 1000 steps. Then two more runs were done with fewer steps for confirmation.

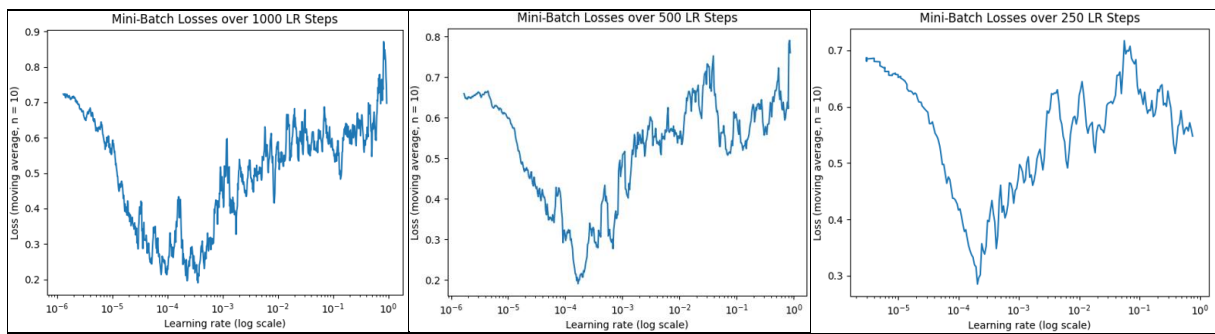


Figure 3.5 – Mini-Batch Loss Fluctuation over Increasing Learning Rate

Figure 3.5 shows a series of steep gradients reaching the minimum loss from the learning rate of 10^{-5} to somewhere between 10^{-4} and 10^{-3} . Instead of pinpointing the optimal learning rate, an initial learning rate of 10^{-3} was chosen. By using polynomial decay, the project attempts to gain quick convergence in the beginning, followed by lower learning rates to obtain local minima. These lower learning rates would eventually cover the values below 10^{-4} .

3.2.4. Loss Function

Xie et al. (2021a) do not specify the loss function used in SegFormer in the original paper. However, in the official PyTorch implementation of SegFormer, the cross-entropy loss is defined as the loss function (Xie et al., 2021b) which is expressed as the following equation:

$$L_{ce} = - \sum_{i=1}^n t_i \log p_i$$

Where t_i is the truth value, p_i is the Softmax probability of the i^{th} class

Cross-entropy loss function was widely used in the studies addressed in the literature review, which is a differentiable function that measures the difference between two probability distributions for the same underlying set of events.

3.2.5. Evaluation Metrics

For this study, Intersection-of-Union (IoU) was employed as the primary evaluation metric. Accuracy was calculated as a supplementary evaluation metric throughout the study, while precision and recall were computed for the top three performing models of each model series (Model-A, Model-G, Model-O, and Model-B1 series). All four metrics are based on the confusion matrix Figure 3.6.

		Predicted class		
		Positive	Negative	
True class	Positive	<i>TP</i>	<i>FN</i>	True positives
	Negative	<i>FP</i>	<i>TN</i>	True negatives
Total		Pred positives	Pred negatives	

Figure 3.6 – Confusion Matrix

IoU or Jaccard index is a metric which can be used to quantitatively assess the degree of overlap between the predicted segmentation and the ground truth segmentation, which is expressed as the following equation:

$$IoU = \frac{TP}{(TP + FP + FN)}$$

Most models reviewed during the literature review demonstrated their model’s effectiveness using IoU, or mean IoU (mIoU) for multiclass semantic segmentation problems. This mIoU should not be confused with the average IoU score throughout this paper. The IoU score of a model, unless otherwise specified, will refer to the average of all the IoU scores for each image or mini-batch. A mIoU score refers to the mean of IoU per class, which will not apply to this *binary* semantic segmentation study. F1-score and dice coefficient are also meaningful metrics, however, as they are positively correlated to IoU, there is little point to report them alongside IoU, unless the Dice coefficient was employed as the loss function. Even though IoU can fail to emphasize over- and under-segmentation properties

(Zhang et al., 2021), IoU remains to be a popular evaluation metric for its intuitiveness, simplicity, and effectiveness.

Accuracy is a metric used to represent the general correctness of the classified pixels. This evaluation metric was used with caution as discussed in section 3.2.6. Training. The metric is expressed with the following equation:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Both precision and recall assess the models based on *relevance*, however, precision represents the fraction of the relevant instances among all the instances predicted to be relevant, whereas recall evaluates the model based on the correctly identified relevant cases among all the relevant cases.

Precision and recall are expressed by the following equations:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

As such, precision would be a recommended metric to evaluate a model if the cost of a False Positive is high, whereas recall would be useful when the cost of a False Negative is high.

3.2.6. Training

This study has experimented with 3 different models based on the SegFormer encoder model MiT-B0 and one MiT-B1. In all cases, a batch size of 2 was used due to the VRAM limitation. However, gradient accumulation was used with an accumulation step of 8 to match the batch size used in the original SegFormer study (Xie et al., 2021a, p. 6). Every train epoch was followed by a validation epoch. Accuracies and cross-entropy losses were observed for both cases, however, the IoU score was calculated only for the validation dataset. This was due to the computing power limitation, as IoU calculation increased the training time by an hour (about a 50% increase). The accuracy was observed

as a supplementary metric to approximate the model's performance. Indeed, most studies encountered during the literature review reported precision and/or recall, rather than accuracy. However, this project is a special case in that the cost of false negatives and false positives is yet to be determined. Recall favours capturing as many true positives as possible, at the cost of increasing false positives. This could be problematic when the model gets integrated with the SDDS, where the sharpness of the segmented region will affect the weight that will be given to the defect identified. On the other hand, the defect detection model that will be used together with the segmentation model can potentially mitigate these false positives by not detecting defects at all in those regions. Hence, until the effectiveness of the defect detection model, as well as its interaction with the segmentation model is observed, the cost of false negatives and false positives is hard to be estimated. Hence, the author assumed a neutral stance and selected accuracy.

To decide on the appropriateness of using accuracy as a supplementary score, the class balance was calculated by counting the pixels of the mask images. For Mixed Dataset, 27.07% of the mask image's pixels were annotated as 1 (3DP object), on average.³ Type 1 data's case was 23.04% and the validation dataset's case was 24.29%. These ratios are classified as "mild imbalance" (Google Developers, 2022), meaning the accuracy score can be relevant. This was done to quickly approximate the binary class balance of the dataset, hence more advanced methods such as Shannon entropy were *not* considered. Regardless, the top-performing models were to be selected based on their IoU performance on the validation dataset, and precision and recall were calculated for the top three IoU performance models at the end.

Model-A was trained for 11 epochs when it met the initial aim of reaching a 0.85 IOU score. At this point, aiming to enhance the robustness of the model, random transformations of Flip, RandomRotate90, Rotate, a minor ColorJitter (up to 0.05 for brightness, and contrast), and hue transformation with a value of 0.1 were applied to the dataset at the data loader level. This approach

³ As all the image dimensions are identical, the global ratio of class 0 and 1 is the same as the average.

modifies the original Mixed Dataset differently every epoch, leading to further data augmentation without requiring more VRAM. The project opted to simulate transfer learning with this approach. The learning rate decay schedule was halted for the first 3 epochs to allow larger model changes. Furthermore, as the convergence almost flattened at a learning rate of $2e-5$, hence from epoch 54, the learning rate was fixed at $2e-5$. After epoch 57, the training was stopped as the validation convergence rate and the IoU score increase rate became very subtle, as shown in Figure 3.7.

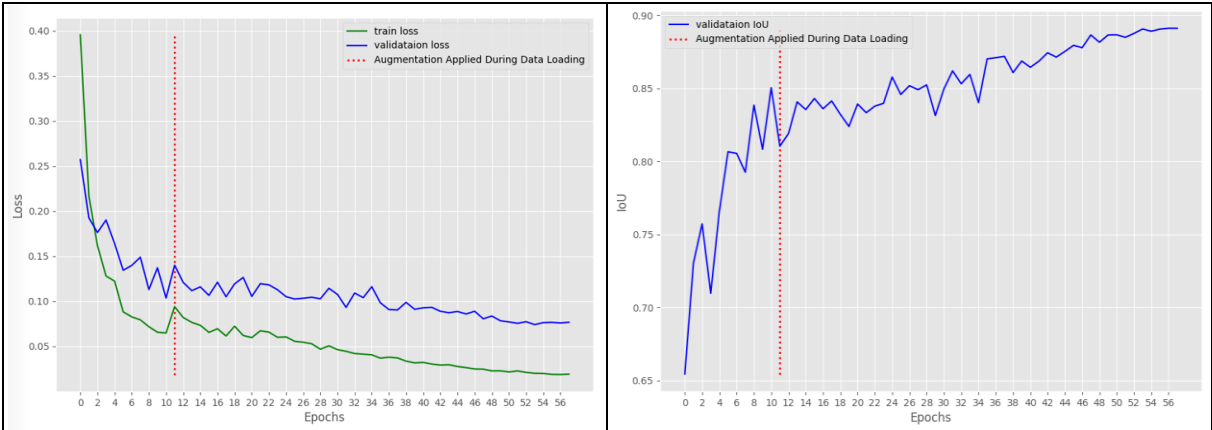


Figure 3.7 – Model-A Convergence and Validation IoU over 58 Epochs

After epoch 17, Model-G was created as an experiment to see the model’s performance on grayscale images. The intuition was that, since 3DP objects can have a very wide range of colours, training with grayscale images could boost the performance of the model. Ceteris paribus, both the training and the validation data were converted to grayscale and the model was continued to be trained, again halting the learning rate decay for the first 3 epochs. The training was halted when it was noticed the model could not reach the pre-grayscale best score even after 23 epochs of training, as shown in Figure 3.8 below.



Figure 3.8 – Model-G Convergence and Validation IoU over 42 Epochs

Model-O is a model trained only on Type 1 Data with Flip and Rotate augmentations at the data loader level. For this case, however, the augmentation was done from the beginning, at epoch 0. This experiment was conducted as the dataset creation was a core aspect of this study. The author attempted to see whether the original data size and variety (Type 1) were sufficient for the model to achieve acceptable generalization. The training was halted as the training loss and validation loss diverged and the IoU increase rate became very subtle as shown in Figure 3.9.

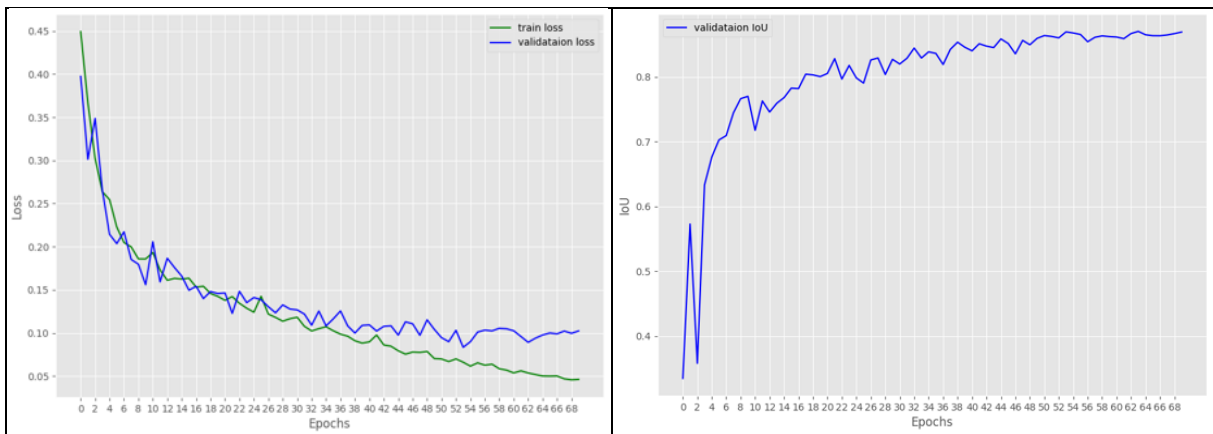


Figure 3.9 – Model-O Convergence and Validation IoU over 70 Epochs

Lastly, Model-B1 was trained on Mixed Dataset. The same data loader-level random transformations applied to Model-A from epoch 11 were applied to this experiment from epoch 0. However, it uses the substantially heavier SegFormer MiT-B1 model instead of MiT-B0. MiT-B1 has 13.1 million parameters for the encoder and 0.6 for the decoder, as opposed to MiT-B0's 3.4 and 0.4 parameters, respectively. As a result, the number of flops for 1024x1024 images is much higher as well with 243.7 million, almost double the MiT-B0's 125.5. The purpose of the experiment was inspired by

the result presented by Xie et al. (2021a, p. 7), where MiT-B1 scored about 0.02 higher mIoU score compared to MiT-B0. This model was later discarded completely, as will be discussed in Section 4. Results and Discussion.

4. RESULTS AND DISCUSSION

Table 4.1 summarizes the performance metrics of the top 3 models, based on the IoU score, found for Model-A, Model-G, and Model-O series. The train epochs should not be compared against other model series, as the per-epoch train time differs from each other.

Model	Train Epoch	IoU	IoU σ	Accuracy	Precision	Recall
Model-A1	56/57	0.8931	0.1609	0.9732	0.9601	0.9638
Model-A2	53/57	0.8931	0.1601	0.9734	0.9588	0.9649
Model-A3	57/57	0.8924	0.1635	0.9733	0.9604	0.9625
Model-G1	41/41	0.8458	0.1837	0.9619	0.9441	0.9446
Model-G2	40/41	0.8447	0.1751	0.9613	0.9377	0.9502
Model-G3	36/41	0.8432	0.1786	0.9592	0.9332	0.9531
Model-O1	63/69	0.8704	0.1787	0.9675	0.9537	0.9571
Model-O2	53/69	0.8697	0.1762	0.9674	0.9543	0.9513
Model-O3	69/69	0.8694	0.1823	0.9656	0.9531	0.9564

Table 4.1 – Top 3 models for each experiment based on the average IoU score.

Model-A series was intended to be the project’s main model. It was previously hypothesized, with added diversity via synthetic data creation, the model would perform significantly better than the model trained only on the original data (Type 1). The Model-A series indeed scored the highest across the evaluation metrics, however, the difference against the Model-O series was smaller than expected. Still, although it cannot be directly compared as this project is a binary segmentation study, Xie et al.’s (2021a) SegFormer achieved a 0.02 improvement in the mIoU score by nearly quadrupling the number of parameters of the model (MiT-B0 vs. MiT-B1). Hence, this is still a significant improvement. Furthermore, Model-O’s results also indicate the original 3DP objects dataset created by this project is diverse and large enough to develop a decently effective model with SegFormer.

Before the in-depth evaluation of inference results, both Model-G and Model-B1 series were discarded. The experiment with Model-G has indicated colour information still benefits the model for predicting and generalizing 3DP objects, even though there is essentially no limit to 3D print filament colour variety. A possible explanation is that despite the wide range of available filament colours, some

colours are simply much more popular than others. For example, white, red, blue, green, and black were much more popular in the dataset than pink or light grey. It could be meaningful to study the correlation between the IoU performance and the colour of 3DP objects. This model was trained significantly less than Model-A and Model-O, but as shown in section 3.2.4 the model had trouble improving the metrics back to pre-grayscale performance even after 45 hours of training.

Model-B1 was discarded after about 39 hours of training (13 epochs), due to the long training time and inferencing time on the CPU. With an NVIDIA T4, its average training time based on the first 5 epochs was 187 minutes per epoch and the validation time was 449 seconds per epoch. In comparison, the MiT-B0 model resulted in 116 minutes and 433 seconds, respectively. MiT-B1's average inferencing time with Intel(R) Xeon(R) (2.20GHz) was 8.39 seconds per image, as opposed to MiT-B0's 5.85 seconds. As the project aims to build a system that runs on a wide range of laptops, the CPU inference time is much more important than the GPU time. Furthermore, it is anticipated at least 10 images will need to be taken and processed in 4 different camera positions, the final difference will accumulate to about 102 seconds. Therefore, this model was discarded. The training results are provided in Appendix D as a reference, without further analysis. Nonetheless, a more serious issue is the RAM requirement of MiT-B1. MiT-B0 already requires about 3.5GB of free RAM during inferencing which is very close to the target RAM usage limit of 4GB. This limitation was set since the final imaging system may be designed for a Raspberry Pi 5 as well, which starts at 4GB RAM. Nevertheless, 8GB RAM is prevalent for laptops, and Raspberry Pi's RAM can also be upgraded. Hence, depending on the further development plan, it may be wise to study the capabilities of MiT-B1.

Model-A and Model-O generally shared similar weaknesses. The first weakness observed was, as anticipated previously, both models had difficulties in segmenting 3DP objects that contain multiple colours as illustrated in Figure 4.1. Nevertheless, it was observed their behaviour was visibly different depending on the way different colours are present. In the case of discrete colour separation, as illustrated in the first column in Figure 4.1, Model-A2 performed significantly worse than Model-O1. Model-A2 has segmented most of the red hues while missing out on most of the purple hues. In

contrast, Model-O1 managed to segment most of the 3DP object. Nevertheless, it does not appear that Model-A2 has simply correlated red hues to be a much stronger characteristic of a 3DP object. The second column also presents a case where both red and purple hues are visible, but Model-A2 performs better than Model-O1 in this case.




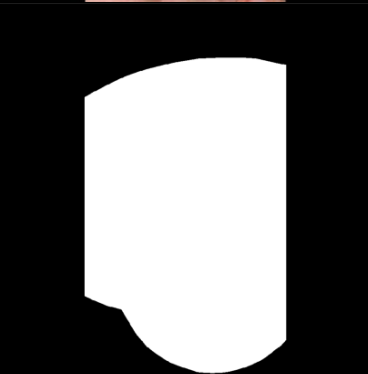
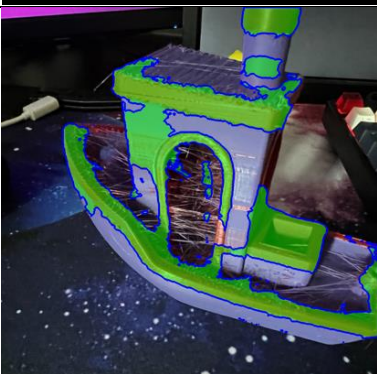
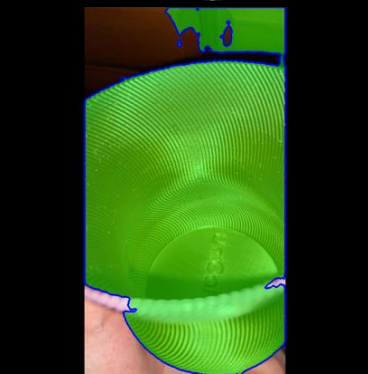
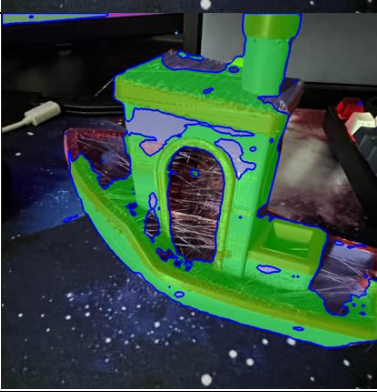
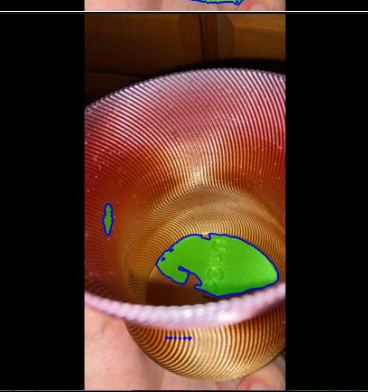
Image						
Ground Truth						
Model-A2						
Model-O1						

Figure 4.1 – Inference Results on Multi-Colour Filaments

It can be speculated that Model-A2 has learned to correlate a sudden change in the colour signal with object edges and treat the pixels outside the most probable 3DP object pixel to be the background. Where the colour signal changes gradually, the model appears to recognize the continuity of the object. A question remains though, as the model has decided to segment only the red part while disregarding most of the purple area in the first column image. Upon a further inspection of the image with the original 1024x1024 resolution, it was found the slicer layer pattern is more clearly visible in the red area than in the purple area. Still, it is true the purple area also exhibits some slicer layer patterns, albeit proportionally less regarding the total purple area. The author can hypothesize there may be a certain threshold for this proportion, but with very little data on the multi-coloured 3DP objects, no conclusion can be drawn in this regard. What could be concluded, though, is that Model-A2 is more practical than Model-O2 as it is rather rare for FDM printer users to use multi-colour filaments in such a discrete (as opposed to gradient) fashion, as many opt to print an object with one colour and then paint over it instead.

Furthermore, the featureless area also affected the models differently. For Model-A2, a motion blur appears to affect the performance less than it affects Model-O1 as illustrated in Appendix A, the first column. However, when the object itself lacks features, such as the slicer layer pattern, the segmentation is poor as shown in the second column. This was expected behaviour, as a 3DP object is indistinguishable from general plastic objects when such features are missing. However, with better lighting, such as in the imaging system developed during this project, the visibility of features should increase. If the object is completely flat, and barely exhibits these features even with adequate lighting, that is highly likely to be an indication of a successful print with very consistent extrusion and without visible defects. Therefore, the superior segmentation result of Model-O1 in the second column does not necessarily mean it is a better model than Model-A2 for this project's purpose. Of course, there could be a use of this different behaviour further into the development of the defect detection system. On the other hand, its poorer performance in the case of motion blur also does not mean it is significantly less useful than Model-A2. Motion blurs are unlikely to occur during imaging,

even though this system aims to segment 3DP objects on a conveyor belt—3DP objects remain almost completely stationary for several minutes to hours when the next print is being extruded. Although, with the structurally unstable setup of the printer itself, there can be an issue of vibration. However, if the stability of the printer is so low that it causes such a significant vibration, the printer will produce a very poor-quality object which will be identified at the first stage of the defect detection system.

Another weakness in the models’ performances was found on 3DP objects with severely distorted layer patterns, but in a rare way. The first column in Appendix B shows a case of ringing defect caused by the printer vibration; poor edges at the arch probably due to lack of support; and inconsistent extrusion in two different ways--over-extrusion and over-heating. With such a mix of multiple issues, the object exhibits a unique print pattern. Even so, Model-A2 does an adequate job, whereas Model-O1’s performance is subpar. In comparison to the first column, the print pattern of the 3DP object in the second column is also severely distorted. However, the only issue here is an under-extrusion problem, which was abundant in the train data. As a result, both Model-A2 and Model-O1 produced highly accurate inference results.

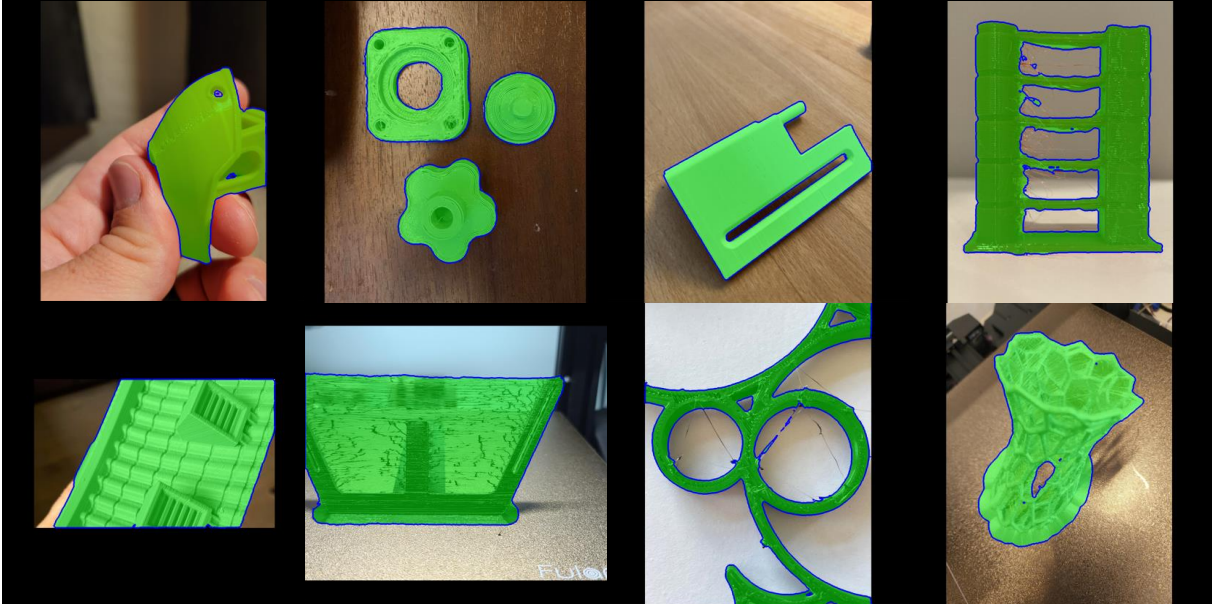


Figure 4.2 – A Sample of Model-A2 Inference Results

Besides these issues, the models produced impressive results even for relatively complex shaped objects. Since it was established that Model-A2 performs better and is more appropriate for

this project's aim, Model-A2's results are illustrated in Figure 4.2. The model still struggles to exclude holes from the segmented regions. However, this problem was consistently noticed only where some stringing defects covered the holes. If the hole is completely clear of 3DP filaments, usually the model was successful at omitting the holes from the segmentation. This issue is likely to stem from the fact that stringing defects often layered over solid 3D printed areas, hence during the annotation phase the strings were included within the annotation polygon drawn.

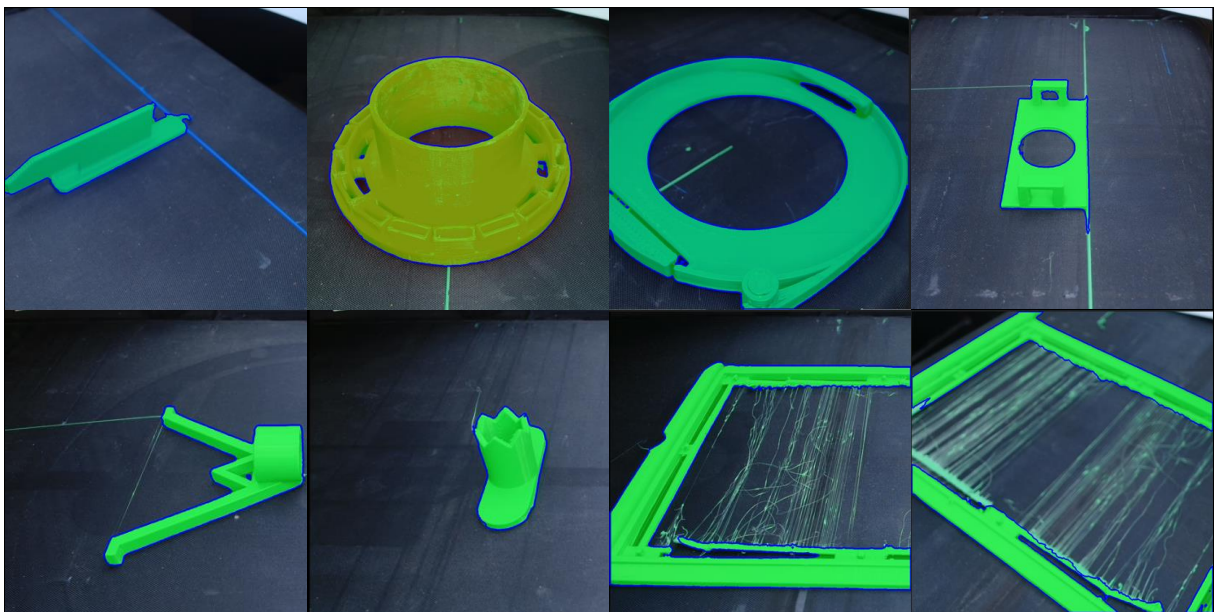


Figure 4.3 – Inference Results in the Imaging System Environment

Lastly, some tests were conducted in a controlled setup in the imaging system with Model-A2. Due to the budget reason, only a handful of objects were printed to be tested. The evaluation metrics were not computed for this test, since the metrics based on a very low number of test objects will not hold a significant meaning with a potentially severe bias. However, it was established in the previous discussions that the shape and colour of the object appear to have less impact on the performance of the model than the presence of 3DP features. Thus, the performance was assessed only qualitatively based on the inference results. In this test, Model-A2 produced an impressive result as shown in Figure 4.3. An unexpected outcome was noted as even highly blurry parts of the print were segmented with high accuracy as shown in Appendix C. The author could only hypothesize this is due to some images taken from the imaging system being included in the training data. Still, only 12 out of the entire dataset was taken from the imaging system, which appears to be too little to fully explain this

behaviour. This should not be a significant problem when the imaging environment is restricted only to a controlled imaging system such as this. As previously discussed, when the imaging is done in multiple different focus values in the SDDS, the defect detection model's inference on these highly blurry areas should receive much lower weight due to the low sharpness than the defects detected in high sharpness images. Overall, further study of the segmentation model's behaviour is required, but that needs to be done together with the planned defect detection model to comprehensively balance the problems arising from each model while observing their interactions.

5. CONCLUSION

This project was conducted to establish the foundation for a surface defect detection system for 3D-printed objects. The project has achieved four main accomplishments. First, a functional prototype of an automated imaging system has been built based on the 2-stage defect detection system proposed, which can consistently capture clear images from 270 degrees of flexible angles. Nonetheless, perhaps the most significant contribution of this study is the creation of the first pixel-wise 3DP object segmentation dataset which can be used for semantic segmentation tasks. The diversity and the size of the dataset's sufficiency were shown by Model-O1's IoU score of 0.8704 against the validation dataset, a result which simultaneously shows SegFormer's optimization and generalization capability. Then, a data augmentation method mixed with synthetic data creation was demonstrated while not overwhelming the training data with randomly cropped 3DP object region, which is often insufficient to infer it to be a 3DP object. The final dataset, Mixed Dataset, amounts to a total of 20,300 real images and synthetically created images. Although only 20,300 training data were created using the hybrid method, there is no real limitation to producing more data. However, the potential further increase in the model performance as a result of synthetically increasing the training data more than this amount is yet to be studied. Lastly, Model-A2, the SegFormer MiT-B0 model trained on Mixed Data was found to be our best model, scoring an IoU score of 0.8931, a 0.0227 increase from Model-O. The evaluation of the inference results also indicates Model-A2 is the most appropriate model for the defect detection system.

6. LIMITATIONS AND RECOMMENDATIONS FOR FUTURE WORKS

Some of the study progress was delayed by budgetary limitations. A premium GPU such as NVIDIA A100 was available on google collab. However, although A100 could more than halve the per-epoch training time of NVIDIA T4 (without performance metrics calculation) from around 80 minutes to 35 minutes, its monetary cost was 7 times an NVIDIA T4. Additionally, the per-epoch training time can be reduced further with a different dataset size to maximize the use of the available VRAM. With an A100 and a batch size of 4, only a bit more than half the VRAM was used, as it was not enough for a batch size of 8. The T4's case was similar, with a batch size of 2 and incapable of processing a batch size of 4. This means that if the dataset size is reduced by 10 to 20%, the VRAM could be used more effectively. Albeit VRAM itself is not the main factor in processing time, it appears the GPU cores were not fully exploited as the data transferred to the device (at a batch size of 2) is significantly smaller than the total data the GPU is capable of processing in parallel. Furthermore, instead of using the same Mixed Dataset, it's possible to generate multiple sets of mixed datasets, each of them consisting of the same balance of Type 1, 3, and 4. Then, for each epoch, a different set could be loaded into the device during the training. This method could greatly improve the train data diversity. However, as demonstrated by this study, it is questionable whether the increased diversity through synthetic data creation would significantly improve the model's generalization further from this point when the original dataset alone was found to be already large and diverse enough for the model to achieve a noteworthy generalization.

Another recommendation to accelerate future research is to stop measuring the accuracy metrics for both the training and validation datasets. This study has already highlighted the change in the accuracy, loss and IoU until the point where the fluctuation became very subtle. Furthermore, the reliability of accuracy as a means of performance measurement is often questionable especially in the presence of class imbalance, albeit mild for this project's dataset. Instead, from this point on, the model could be trained with only the validation IoU calculation other than the losses. Then, other

metrics such as precision and recall could be calculated for the top-performing models to save some more training time.

Many aspects of this project’s semantic segmentation study imitate the original SegFormer work of Xie et al. (2021) to simplify the process, especially when setting the hyperparameters. By doing so, this paper has established a benchmark for future work. Different optimizers, learning rate schedules, and batch sizes could be experimented with to see the effect on model convergence.

The model can further benefit from better image annotation. Although the 3DP images were annotated painstakingly, in some cases it was difficult to annotate complex shapes or objects with many holes inside, due to the way polygon annotation works—the hole region cannot simply be removed from a polygon, but the polygon line must begin from outside the object, then inside the hole, and back out, which adds complexity. Figure 6.1 shows an example of such a case. As shown, it requires a lot of polygon points, and an unexpected polygon behaviour may lead to a segmentation mask different from what is visualized in CVAT.

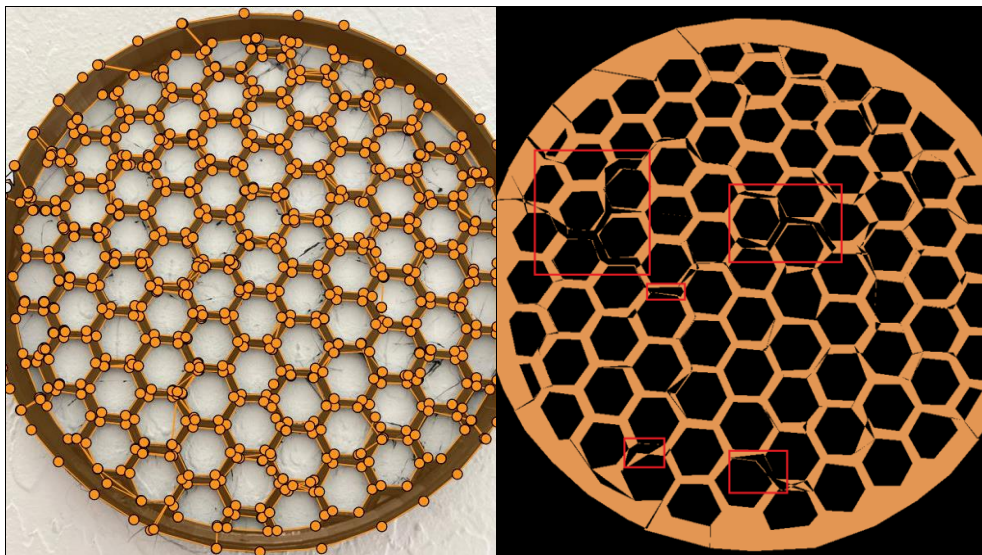


Figure 6.1 – Complex Shape Annotation and Unexpected CVAT Behaviours

Model-B1 could be studied further. Firstly, it was previously noted that Model-B1 was discarded partially due to its long training time. This relates back to the budget issue since the 3-hour-per-epoch training time requires a substantial monetary cost. MiT-B1 having nearly 10 million more

parameters than MiT-B0, could result in a higher IoU score, whereas MiT-B0 seems to plateau at 0.8931. It may be feasible to test further to see whether its IoU performance increase could outweigh the long CPU inference time and higher requirement of RAM.

This study also has established a foundation to build the proposed SDDS. The integration of the segmentation algorithm with the imaging system hardware is planned. Focus-stacking techniques could be adopted for a better user interface, which is already being experimented with by the author. Furthermore, there has been a rapid advancement in photogrammetry in the past few years. Especially the work of Müller et al. (2022) from NVIDIA has drastically reduced the training and rendering time of Neural Radiance Field (NeRF) models. The imaging system developed in this project, with its capability to capture images from 270 degrees angle, could be experimented with their work to see whether it could be used as a 3D scanner. The technology of course still requires a powerful GPU, which would be only usable by power users. Nevertheless, the core aim of future work remains to be the development of a 3DP object SDDS. In the immediate future, bounding box annotation of defects is planned, which will be integrated with the semantic segmentation task done in this project.

7. REFERENCES

- 3DQue. (2022, June 23). *Stop Epic 3D Print Fails With AI - Try it Live!* [Video]. YouTube.
<https://www.youtube.com/watch?v=aNNFyo5-Zmw>
- Bai, Y., Mei, J., Yuille, A. L., & Xie, C. (2021). Are Transformers More Robust Than CNNs? *Advances in Neural Information Processing Systems*, 34, 26831–26843.
<https://doi.org/10.48550/arxiv.2111.05464>
- Bengio, Y., Goodfellow, I., & Courville, A. (2015). *Deep learning* (Vol. 2) [Academia.edu]. Cambridge, MA, USA: MIT press.
- Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Alan, L., & Zhou, Y. (2021). TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation. *ArXiv Preprint*.
<https://doi.org/10.48550/arxiv.2102.04306>
- Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *Computer Vision – ECCV 2018*, 833–851. https://doi.org/10.1007/978-3-030-01234-2_49
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ArXiv: Computer Vision and Pattern Recognition*. <https://doi.org/10.48550/arXiv.2010.11929>
- Fast.ai. (2016). *Fastai Documentation*. <https://docs.fast.ai/callback.schedule.html>
- PCGamer. (2016, December 15). *Amazon slashes price on Logitech C920 webcam and other peripherals*. PCgamer. <https://www.pcgamer.com/amazon-slashes-price-on-logitech-c920-webcam-c920-and-other-peripherals/>
- Ghali, R., Akhloufi, M. A., & Mseddi, W. S. (2022). Deep Learning and Transformer Approaches for UAV-Based Wildfire Detection and Segmentation. *Sensors*, 22(5), 1977.
<https://doi.org/10.3390/s22051977>



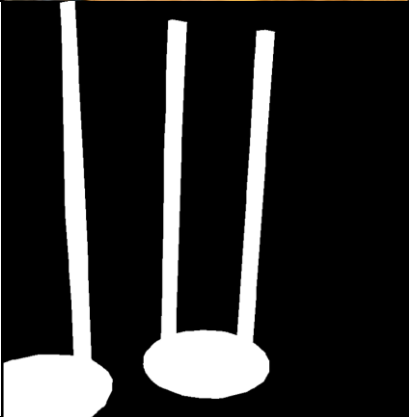

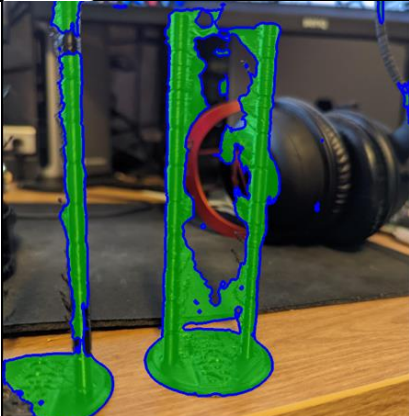
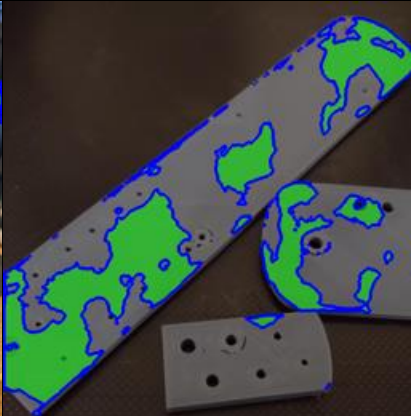
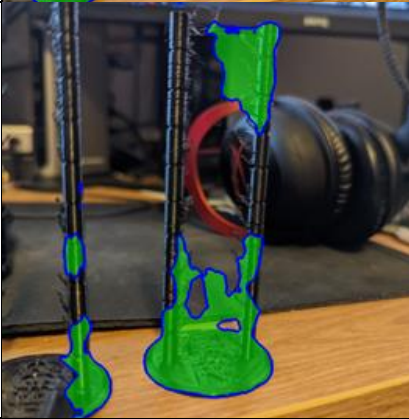
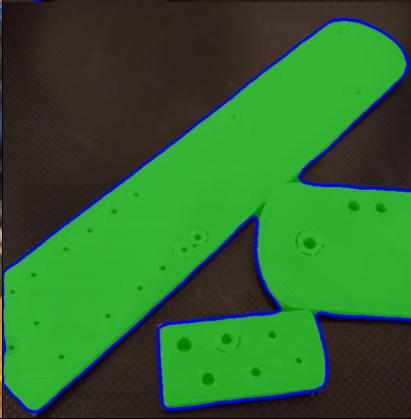
- Google Developers. (2022, July 18). *Imbalanced Data*. Google Developers Foundational Courses.
<https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data>
- Guo, Y., Liu, Y., Georgiou, T., & Lew, M. S. (2017). A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, 7(2), 87–93.
<https://doi.org/10.1007/s13735-017-0141-z>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.
<https://doi.org/10.1109/cvpr.2016.90>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems*, 1, 1097–1105.
<https://doi.org/10.5555/2999134.2999257>
- Lehmann, T., Gonner, C., & Spitzer, K. (1999). Survey: interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, 18(11), 1049–1075.
<https://doi.org/10.1109/42.816070>
- Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
<https://doi.org/10.1109/cvpr.2015.7298965>
- Mahmood, K., Mahmood, R., & van Dijk, M. (2021). On the Robustness of Vision Transformers to Adversarial Examples. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*.
<https://doi.org/10.1109/iccv48922.2021.00774>
- Mech Solution Ltd. (2022). *AEye Assistant* [Software]. <https://cloud3dprint.com/ai-failure-detection/>
- Mo, Y., Wu, Y., Yang, X., Liu, F., & Liao, Y. (2022). Review the state-of-the-art technologies of semantic segmentation based on deep learning. *Neurocomputing*, 493, 626–646.
<https://doi.org/10.1016/j.neucom.2022.01.005>

- Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), 1–15.
<https://doi.org/10.1145/3528223.3530127>
- Obico. (2019). *The Spaghetti Detective* [Software]. <https://www.obico.io/the-spaghetti-detective.html>
- OpenCV. (2022, January 5). *The OpenCV Reference Manual*.
https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html
- P., A. (2022, January 10). *How to Create Synthetic Dataset for Computer Vision (Object Detection)*. GitHub. <https://github.com/alexppppp/synthetic-dataset-object-detection>
- Pinto, F., Torr, P. H., & K Dokania, P. (2022). An Impartial Take to the CNN vs Transformer Robustness Contest. *European Conference on Computer Vision*, 466–480.
<https://doi.org/10.48550/arxiv.2207.11347>
- Pinto, F., Torr, P. H. S., & Dokania, P. K. (2022). An Impartial Take to the CNN vs Transformer Robustness Contest. *European Conference on Computer Vision*, 466–480.
<https://doi.org/10.48550/arxiv.2207.11347>
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28
- Schlemper, J., Oktay, O., Schaap, M., Heinrich, M., Kainz, B., Glocker, B., & Rueckert, D. (2019). Attention gated networks: Learning to leverage salient regions in medical images. *Medical Image Analysis*, 53, 197–207. <https://doi.org/10.1016/j.media.2019.01.012>
- Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv Preprint*. <https://doi.org/10.48550/arXiv.1409.1556>
- Simplify3D. (2019). *Print Quality Guide*. <https://www.simplify3d.com/support/print-quality-troubleshooting/>

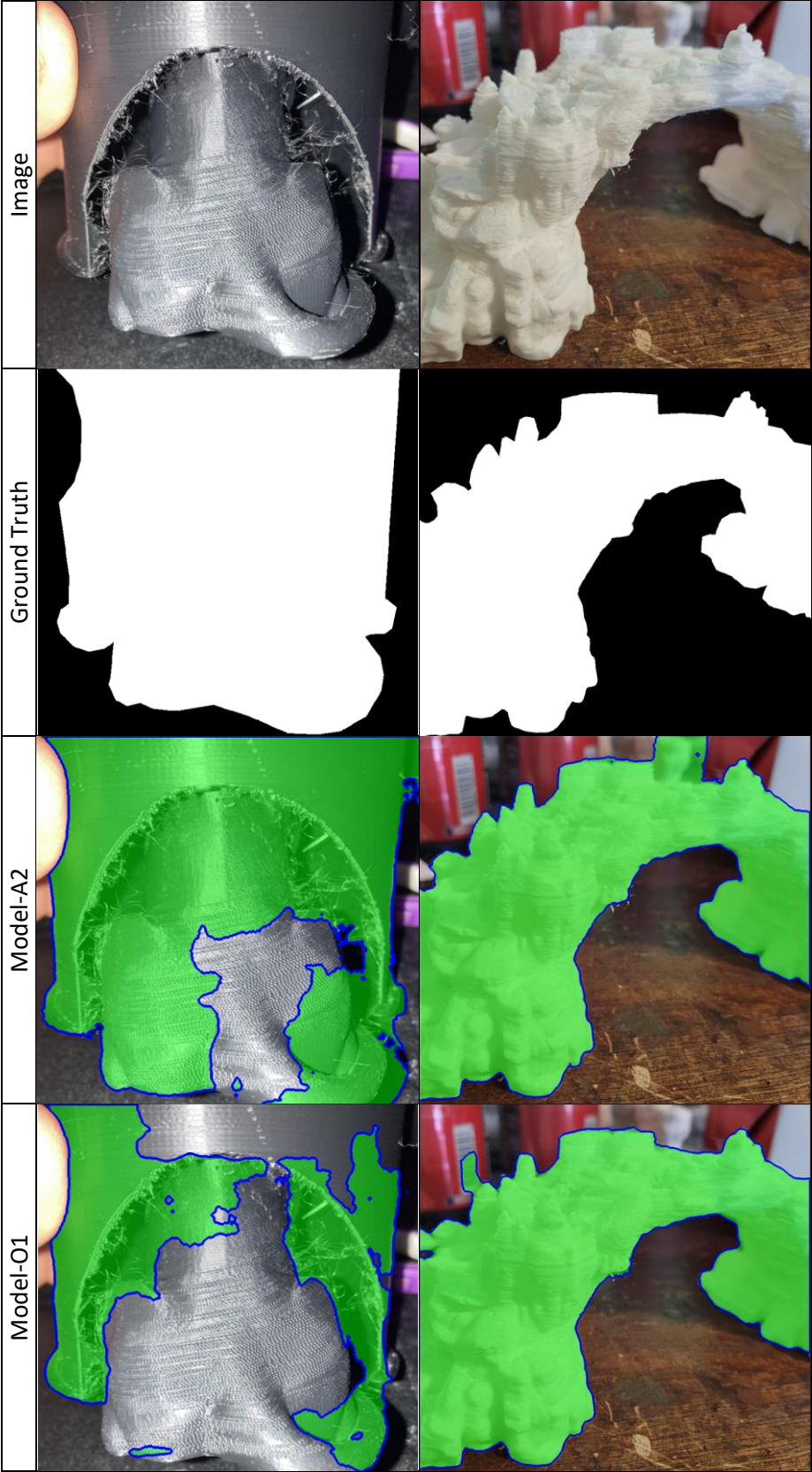
- Smith, L. N. (2017). Cyclical Learning Rates for Training Neural Networks. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*.
<https://doi.org/10.48550/arXiv.1506.01186>
- Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *ArXiv Preprint*.
<https://doi.org/10.48550/arXiv.2207.02696>
- Wang, T., Lan, J., Han, Z., Hu, Z., Huang, Y., Deng, Y., Zhang, H., Wang, J., Chen, M., Jiang, H., Lee, R. G., Gao, Q., Du, M., Tong, T., & Chen, G. (2022). O-Net: A Novel Framework With Deep Fusion of CNN and Transformer for Simultaneous Segmentation and Classification. *Frontiers in Neuroscience*, 16. <https://doi.org/10.3389/fnins.2022.876065>
- Wang, X., Girshick, R., Gupta, A., & He, K. (2018). Non-local Neural Networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7794–7803.
<https://doi.org/10.1109/cvpr.2018.00813>
- Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021a). SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. *Cornell University - ArXiv*.
<https://doi.org/10.48550/arxiv.2105.15203>
- Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021b). *SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers*.
https://github.com/NVlabs/SegFormer/blob/c867a383ffd8285c48ac49d074f4744f490f8ae4/mmseg/models/losses/cross_entropy_loss.py#L139
- Yuan, F., Zhang, L., Xia, X., Wan, B., Huang, Q., & Li, X. (2019). Deep smoke segmentation. *Neurocomputing*, 357, 248–260. <https://doi.org/10.1016/j.neucom.2019.05.011>
- Zhang, Y., Mehta, S., & Caspi, A. (2021). Rethinking Semantic Segmentation Evaluation for Explainability and Model Selection. *ArXiv Preprint*. <https://arxiv.org/pdf/2101.08418.pdf>

- Zheng, Y., Wang, Z., Xu, B., & Niu, Y. (2022). Multi-Scale Semantic Segmentation for Fire Smoke Image Based on Global Information and U-Net. *Electronics*, 11(17), 2718.
<https://doi.org/10.3390/electronics11172718>
- Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2017). Places: A 10 million image database for scene recognition [Dataset]. In *IEEE transactions on pattern analysis and machine intelligence*. IEEE. <https://paperswithcode.com/dataset/places>
- Zhou, D., Xie, E., Xiao, C., Anandkumar, A., Feng, J., & Alvarez, J. M. (2022). Understanding the Robustness in Vision Transformers. *International Conference on Machine Learning*, 27378–27394. <https://proceedings.mlr.press/v162/zhou22m/zhou22m.pdf>

8. APPENDIX A. INFERENCE RESULTS ON LESS VISIBLE 3DP FEATURES

Image		
Ground Truth		
Model-A2		
Model-O1		

9. APPENDIX B. INFERENCE RESULT ON SEVERELY DISTORTED FEATURES

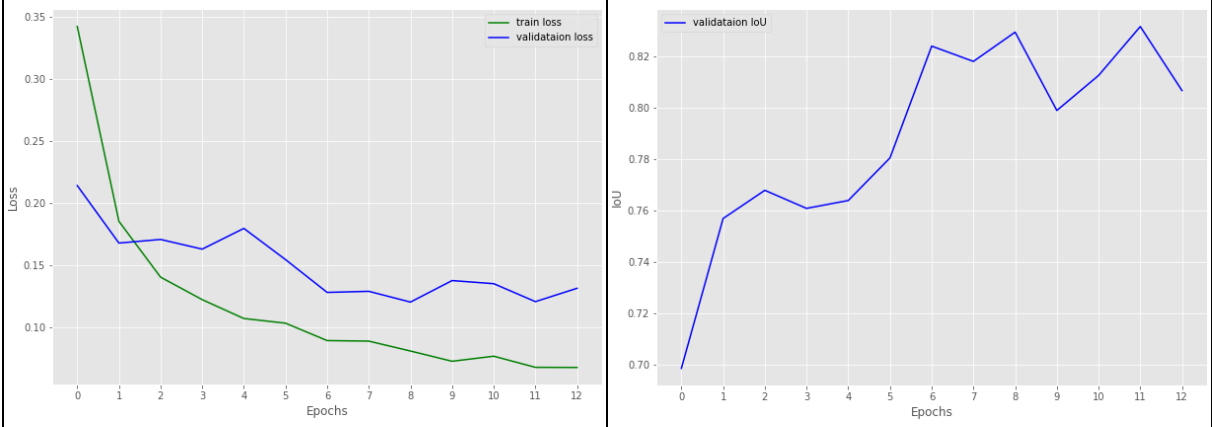


10. APPENDIX C. INFERENCE RESULTS IN THE IMAGING SYSTEM ENVIRONMENT

with increasing focus value from severe under-focus (top) to severe over-focus (bottom)



11. APPENDIX D. MODEL-B1 CONVERGENCE AND VALIDATION IOU OVER 13 EPOCHS





NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa