

## Article

# Synthetic-to-Real Composite Semantic Segmentation in Additive Manufacturing

Aliaksei Petsiuk <sup>1</sup>, Harnoor Singh <sup>1</sup>, Himanshu Dadhwal <sup>1</sup> and Joshua M. Pearce <sup>1,2,\*</sup><sup>1</sup> Department of Electrical & Computer Engineering, Western University, London, ON N6A 3K7, Canada; apetsiuk@uwo.ca (A.P.); hdadhwal@uwo.ca (H.D.)<sup>2</sup> Ivey School of Business, Western University, London, ON N6A 3K7, Canada

\* Correspondence: joshua.pearce@uwo.ca

**Abstract:** The application of computer vision and machine learning methods for semantic segmentation of the structural elements of 3D-printed products in the field of additive manufacturing (AM) can improve real-time failure analysis systems and potentially reduce the number of defects by providing additional tools for in situ corrections. This work demonstrates the possibilities of using physics-based rendering for labeled image dataset generation, as well as image-to-image style transfer capabilities to improve the accuracy of real image segmentation for AM systems. Multi-class semantic segmentation experiments were carried out based on the U-Net model and the cycle generative adversarial network. The test results demonstrated the capacity of this method to detect such structural elements of 3D-printed parts as a top (last printed) layer, infill, shell, and support. A basis for further segmentation system enhancement by utilizing image-to-image style transfer and domain adaptation technologies was also considered. The results indicate that using style transfer as a precursor to domain adaptation can improve real 3D printing image segmentation in situations where a model trained on synthetic data is the only tool available. The mean intersection over union (mIoU) scores for synthetic test datasets included 94.90% for the entire 3D-printed part, 73.33% for the top layer, 78.93% for the infill, 55.31% for the shell, and 69.45% for supports.

**Keywords:** 3D printing; additive manufacturing; g-code segmentation; sim-to-real; semantic segmentation; synthetic data



**Citation:** Petsiuk, A.; Singh, H.; Dadhwal, H.; Pearce, J.M. Synthetic-to-Real Composite Semantic Segmentation in Additive Manufacturing. *J. Manuf. Mater. Process.* **2024**, *8*, 66. <https://doi.org/10.3390/jmmp8020066>

Academic Editor: Jing Shi

Received: 1 February 2024

Revised: 26 March 2024

Accepted: 26 March 2024

Published: 28 March 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

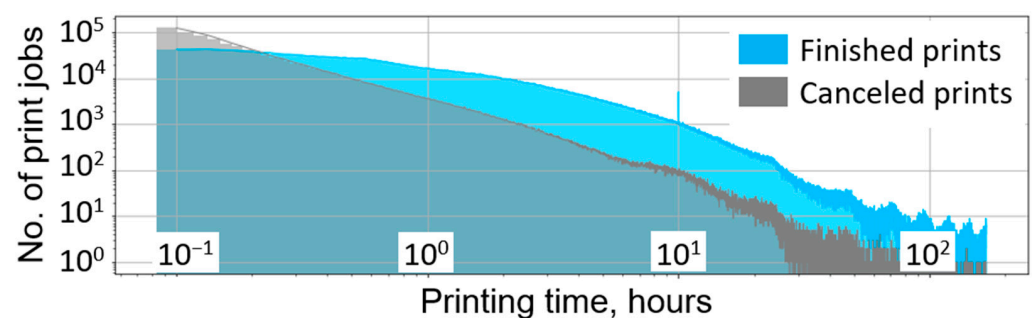
## 1. Introduction

With its current exponential growth, the amount of plastic waste produced could reach 250 billion tons by 2050 [1], vast quantities of which cause pollution of the natural environment on land and in the ocean [2]. Distributed manufacturing using additive manufacturing (AM) is reforming global value chains as its usage increases rapidly [3], because there are millions of free 3D printable consumer product designs and 3D printing them results in substantial cost savings compared to conventionally manufactured commercial products [4,5].

The growing popularity of 3D printing is playing a notable role in the problem of recycling as 3D-printed products rarely have recycling symbols [6], use uncommon polymers [7], and are increasing the overall market of plastic materials [8]. This is not only caused by additional plastic products, but also from disturbing failure rates. Inexperienced 3D printer users are estimated to have failure rates of 20% [9]. Even experienced professionals working in 3D print farms, however, have failure rates of at least 2% [10]. The probability of a manufacturing defect increases with the size and print time of the object (e.g., using large-scale fused filament printers [11] or products [12,13], or fused granule printers [14,15]), which can magnify the waste materials created from even a small percentage of failures. It is clear that the ability to automatically detect deviations in AM will significantly help to reduce material waste and the time spent on reproducing failed prints.

As recent studies [16] show, computer vision is becoming increasingly popular in analyzing AM and extrusion-based 3D printing processes. For example, Ceruti et al. [17] utilized data from computer-aided design (CAD) files that are used in the first step of the design of a 3D-printed component. Then, further down the software toolchain, Nuchitprasitchai et al. [18], Johnson et al. [19], and Hurd [20] developed failure analysis based on comparisons with the Standard Tessellation Language (STL) files used at the slicing step in most 3D printing processes. Further still, both Jeong et al. [21] and Wasserfall et al. [22] used, instead, the G-code files that provide the 3D printer with spatial toolpath instructions for printing parts. The 3D printing software toolchain does not need to be used at all, as several approaches use comparisons with reference data [23,24] or ideal 3D printing processes [25,26]. In addition, a 3D reconstruction-based scanning method for real-time monitoring of AM processes is also possible [27]. In previous works, the authors considered the possibilities of detecting critical manufacturing errors using classical image-processing methods [28], as well as employing synthetic reference images rendered with a physics-based graphics engine [29]. The proposed methods, however, do not fully utilize the available information and are limited in determining the location-based categories of production deviations.

The popular open source Spaghetti Detective application [30,31] is also a direct confirmation of the effectiveness of visual monitoring. An analysis of Spaghetti Detective's [30] user performance database, collected over 2.3 years, showed that 24% of all 5.6 million print jobs were canceled, which can be represented as 456 wasted hours of continuous printing compared to 5232 h of printing where all the print jobs were finished (Figure 1).



**Figure 1.** Analysis of 3D printer users' activity over 2.3 years. The runtime distribution shows a 24% failure rate for all 5.6 million printing tasks longer than 5 min.

This statistic, however, does not include over a million canceled print jobs less than 5 min long, which are assumed to be due to initial bed-leveling issues and cannot, therefore, indicate manufacturing failures. It also does not consider the working time of human operators spent on starting later-canceled printing tasks.

Semantic segmentation [32] of both the entire manufactured part and its separate structural regions at the stage of production of each layer will expand the capacity of the visual analysis of AM processes and can make it possible to more accurately determine the nature of individual production errors depending on their localization. This can provide advanced tools for correcting printing defects in situ, where each successive layer can be modified depending on the deviations found in the previous stage, thus improving both the mechanical and aesthetic characteristics of the entire object. It may also reduce the requirements for camera positioning accuracy and calibration, eliminating the need for visual markers and rigid holders.

In the previous work [29], the authors demonstrated the ability of Blender [33], a free and open-source physics rendering engine, to generate photorealistic images of ideal 3D printing processes based on existing G-code files. This work served as a milestone in the development of a deep learning-based approach, presented in this paper, to the semantic segmentation of structural elements in 3D printing environments.

Using a synthetic dataset, however, comes at the cost of a domain shift, which is often strongly associated with appearance changes [34]. When the source (synthetic images) and target (real images) domains are semantically related, but are different in visual representation, direct propagation of learned knowledge about one domain to another can adversely affect segmentation performance in the latter domain. Therefore, domain adaptation (DA) is needed in order to learn generalized segmentation rules in the presence of a gap between the source and target dataset distributions [34,35].

There are examples in the literature of successful synthetic-to-real (sim-to-real) DA applications. Imbusch, Schwarz, and Behnke [36] proposed an unsupervised Generative Adversarial Network (GAN)-based DA approach to a robotics environment image dataset that provides a performance close to those trained on real data and does not require annotations of real robotic setups. Li et al. [37] presented a semantically aware GAN-based neural network model for virtual-to-real urban scene adaptation with the ability to store important semantic information. Lee et al. [38] introduced a sim-to-real vehicle identification technique consisting of DA and semi-supervised learning methods.

Domain adaptation, however, is a separate area of research and is not covered in this article. The possibility of applying a cycle-consistent adversarial network (CycleGAN) [39]—an image-to-image style transfer method—was considered for segmentation improvement, as generative adversarial networks can perform a significant role in domain adaptation techniques and be used in future research.

The proposed method is a novel approach to segmenting key regions of manufactured parts during their fabrication using G-code information and synthetic data. Revealing this system to end users will allow constant expansion of the synthetic image database for subsequent neural network training. The presented contributions, therefore, can be summarized as follows:

- A technique for generating synthetic image-mask pairs of layer-by-layer ideal 3D printing processes has been developed for subsequent neural network training;
- Three independent labeled synthetic image datasets for (a) the entire part, (b) the top (last printed) layer, and (c) the infill, shell, and supports for 3D-printed objects have been created;
- A neural network was trained for the semantic segmentation of the entire printed part, as well as its last printed top layer and internal structure;
- Image-to-image style transfer approaches to improve segmentation results have been explored.

All the above steps are sequentially described in this article after first reviewing related works in detail. The Results section will discuss the potential for the localization of 3D-printed parts in the image frame, as well as the application of image processing methods to the parts' structural elements for subsequent detection of manufacturing deviations.

## 2. Background

Semantic image segmentation problems represent an actively developing area of research in deep machine learning [32,40]. The main limiting factor, however, is the difficulty of obtaining annotated databases for the training of machine learning architectures. This approach requires thousands of images with labeled masked regions, which is a difficult and time-consuming task—manual annotation of a single image with pixel-by-pixel semantic labels can take more than 1.5 h [41].

The use of synthetic images, in turn, allows the procurement of a segmented training database conditionally “free of charge”, since masked regions of interest can be automatically annotated when creating virtual physics-based renders. In addition, advances in computer graphics make it possible to generate an almost unlimited amount of labeled data by varying environmental parameters in ranges that are difficult to obtain in real conditions [34]. The success of simulated labeled data is clearly illustrated in the already classic GTA5 [42] and Synthia [43] image sets.

There are many examples of applying synthetic datasets to solve real-world practical problems. Nikolenko [44] presented an up-to-date technological slice of the use of synthetic data in a wide variety of deep learning tasks. Melo et al. [45] outlined the most promising approaches to integrating synthesized data into deep learning pipelines. Ward, Moghadam, and Hudson [46] used a real plant leaf dataset augmented with rendered images—for instance, leaf segmentation—to measure complex phenotypic traits in agricultural sustainability problems. Boikov et al. [47] presented a methodology for steel defect recognition in automated production control systems based on synthesized image data.

Several researchers introduced artificial intelligence (AI)-based methods into the AM field to classify the quality of manufacturing regions, as well as to segment failed areas in 3D printing processes. Valizadeh and Wolff [48] provided a comprehensive overview of neural network applications to several aspects of AM processes. Banadaki et al. [49] proposed a convolutional neural network (CNN)-based automated system for assessing surface quality and internal defects in AM processes. The model is trained on captured images during material layering at various speeds and temperatures, and demonstrates 94% accuracy in five failure gradations in real time. Saluja et al. [50] utilized deep learning algorithms to develop a warping detection system. Their method extracts the layered corners of printed components and identifies warpage with 99.3% accuracy. Jin et al. [51] presented a novel CNN-based method incorporating strain to measure and predict four levels of delamination conditions. These works solve a set of specific production problems. The developed algorithms, however, are difficult to generalize and scale. Brion and Pattison [52] introduced an error detection and correction system based on visual and neural network analyses of extruded segments. This study demonstrates promising results; however, it is limited in providing general information about the whole working volume.

Analysis based on semantic segmentation, in turn, has significant potential for detecting and evaluating a wide range of manufacturing defects. Wong et al. [53] have demonstrated U-Net CNN 3D volumetric segmentation in AM using medical imaging techniques to automatically detect defects in X-ray-computed tomography images of specimens. Cannizzaro et al. [54] proposed an AI in-situ emerging defects monitoring system utilizing automatic GAN-based synthetic image generation to augment the training dataset. These functions are built into a holistic distributed AM platform that allows storage and integration of data at all manufacturing stages. Davtalab et al. [55] presented a neural network-automated system of semantic pixel-wise segmentation, based on one million images, to detect defects in 3D-printed layers.

Combining various analysis techniques with the segmentation of characteristic areas of fabricated parts will make a significant contribution to the field of AM. Having an open-structure annotated database for additive manufacturing will create considerable opportunities for the development of failure detection systems in the future. Segmentation and localization of the individual structural elements of manufactured objects can make it easier to detect and track erroneous regions when they occur.

### 3. Methods

Preparation for 3D printing involves layer-by-layer slicing of the model, where each extruded segment corresponds to a certain set of characteristics, such as fan speed, temperature, plastic flow rate, line type (internal, external, and overhang perimeters; support and its interface; solid and internal infill; etc.), reflected in the G-code. By using this information as input to the developed visual processing pipeline, it is possible to create an individual pixel-perfect mask for each section of the manufactured part.

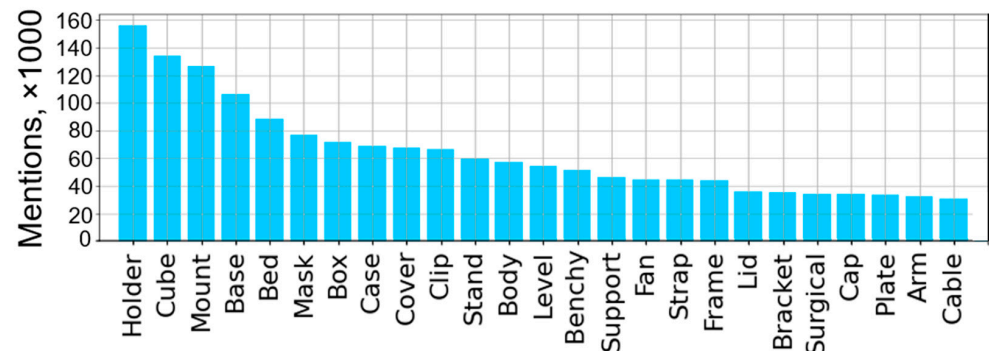
Based on the most common words in 3D print filenames stored in the Spaghetti Detective database [30], sets of labeled images of printed products at various stages of their production were generated in the physics-based graphics engine [29]. These image-mask pairs were further used to train neural networks for the tasks of visual segmentation of manufactured parts and their structural elements. Additionally, the possibilities of image-to-image style translation were also explored, to reduce the domain gap and increase the



segmentation precision. The segmentation efficiency was tested both on synthetic renders outside of training sets and on real images. Data and source code for this project can be obtained from the Open Science Framework (OSF) repository [56].

### 3.1. Creation of Synthetic Image Datasets

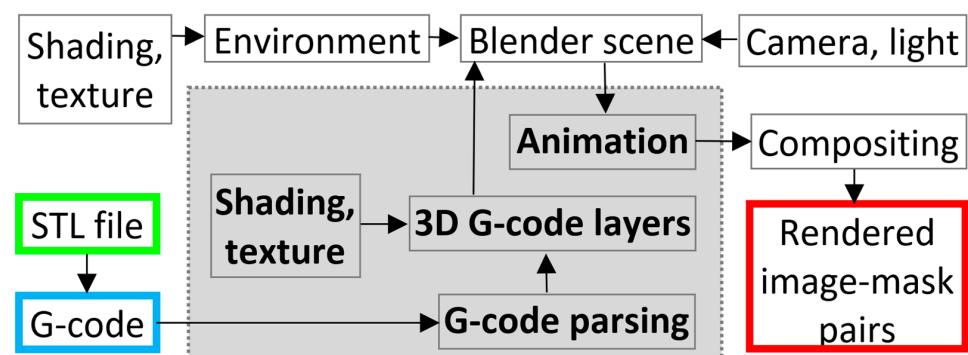
(1) Selecting CAD designs for rendering: More than 5.6 million filenames were partitioned into meaningful lexical parts and processed through Spaghetti Detective's user performance database [30] to create a dictionary of the most frequently used words (Figure 2). These print jobs were performed by 49,000 unique users on 57,000 different 3D printers. The average print time was 3.6 h.



**Figure 2.** Distribution of the 25 most frequently used words in file names for 3D printing. A detailed analysis of the users' print task database is given in the source file repository [56].

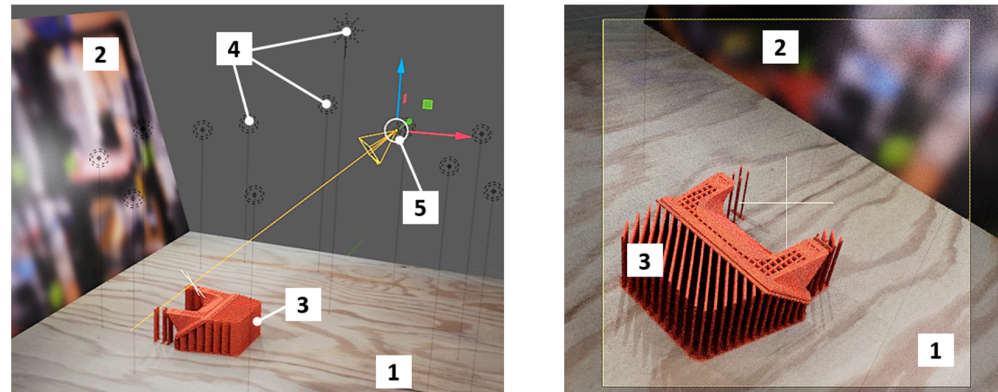
Based on the compiled dictionary, a set of random Standard Tessellation Language (STL) files was collected from Thingiverse [57]—an open catalog of widely used computer-aided designs (CADs) for 3D printing—for further processing. These files formed the basis for generating a database of synthetic images. A complete list of the used CAD designs is in the OSF repository [56].

(2) Graphics rendering pipeline: All the selected STL files were converted into G-codes in free MatterControl software [58], maintaining the same slicing parameters: 0.3 mm layer height, 0.4 mm nozzle diameter, 4 perimeters, and 30% grid infill. The resulting G-codes were further parsed layer by layer in the Blender [33] programming interface, where the extruder trajectory is converted into a set of curves with a controllable thickness parameter and preset material settings. Each G-code layer is thus transformed into an independent 3D object. The whole rendering process is illustrated in the following diagram (Figure 3).



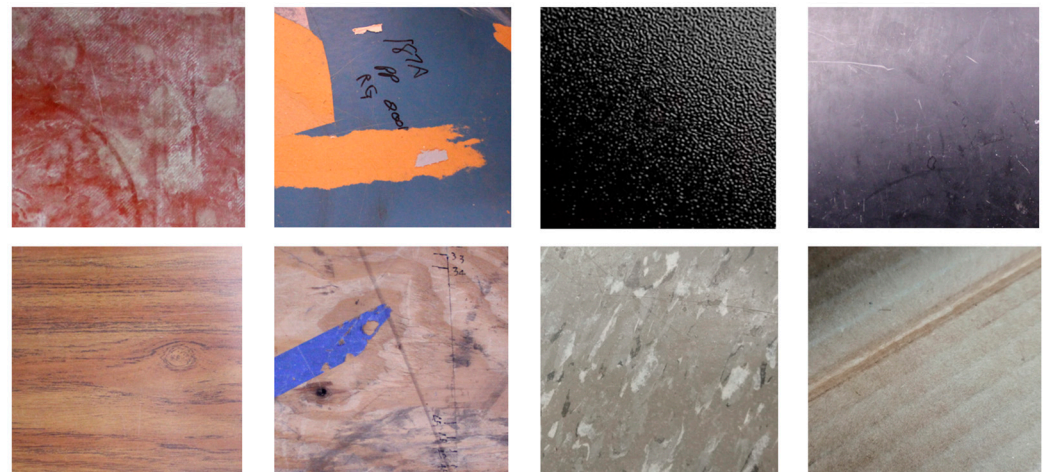
**Figure 3.** Synthetic AM database creation pipeline. Each 3D part, in the form of an STL (green) file, is converted into a set of printer tool head trajectories (G-code, blue), which is the input parameter of the automated scripted section (gray). Blender environment (textures, camera, lights) and compositing settings can also be automated in the future. The image-mask pairs (red) are the result of frame-by-frame animation rendering for each individual G-code file.

The functional component of the repository [59] was used as a basis for importing G-code files into the graphics engine. To create photorealistic renders, scenes similar to real physical environments were created in Blender. The position of the camera, as well as the degree of illumination and the locations of light sources, were chosen to closely match the actual workspace (Figure 4).



**Figure 4.** Blender scene—user window (left) and virtual camera viewport (right); 1—printing bed/ground surface texture, 2—background image plane simulating ambient environment, 3—rendered manufactured part, 4—light sources with variable locations, 5—camera with variable location.

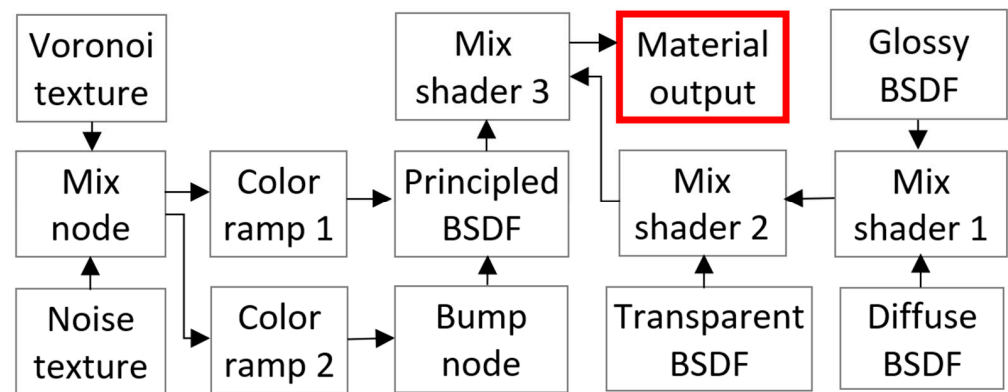
The whole scene, in addition to the printed part, includes components such as point light sources to create diverse heterogeneous all-round illumination; the “Sun”, to create uniform background lighting; a flat printing surface with realistic texture and reflectivity; and a plane with a superimposed blurred background image to create an illusion of a defocused ambient environment. Figure 5 illustrates several examples of realistic textures for the printing bed/ground surface plane.



**Figure 5.** Texture samples for the printing bed/ground surface. More than 15 photographs of surfaces such as wood, metal, paper, stone, and others were superimposed onto the virtual working area. Variations in lighting, cropping, scaling, and image orientation during animation allow the creation of unique backgrounds.

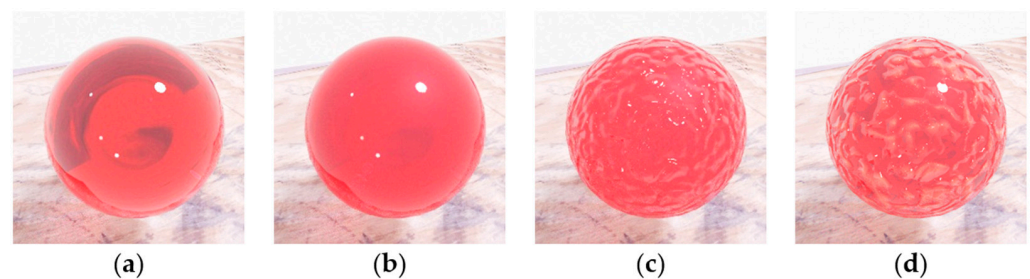
The color of the plastic material and the surface characteristics of the printed part were created and adjusted empirically using a rich library of Blender shaders [60]. When simulating surface irregularities, the Noise Texture [61] and Voronoi Texture [62] nodes were used to add Perlin and Worley noises, respectively, while the “Bump” node was added to adjust the overall roughness. The Mix node was used to balance the Voronoi and Noise textures to create the desired roughness characteristics. Photorealistic color, transparency,

and reflection parameters were obtained through the combination of Principled [63] (adds multiple layers to vary color, reflection, sheen, transmission, and other parameters), Glossy [64] (adds reflection with microfacet distribution), Diffuse [65] (adds Lambertian and Oren–Nayar diffuse reflections), and Transparent [66] (adds transparency without refraction) Bidirectional Scattering Distribution Functions (BSDFs) (Figure 6). Mix shaders 1 to 3 were used to adjust the strength of each BSDF component in the material output.



**Figure 6.** The shading node network has been experimentally developed to achieve maximum realism of generated renders. The creation of all connections and node settings is fully automated in the code, which provides the flexibility to adjust the color, transparency, reflectivity, and other characteristics of the output material (red).

A detailed example of the creation of such a texture is shown in Figure 7. The Principled BSDF node represents an elementary material. Adding Diffuse, Glossy, and Transparent shaders allows material variations to create a desirable effect. Adding different types of noise can simulate realistic unevenness and deviations in elevation in a surface map. The provided selection and hierarchy of BSDF shaders were chosen experimentally, and the desired result may be achieved in multiple alternative ways.

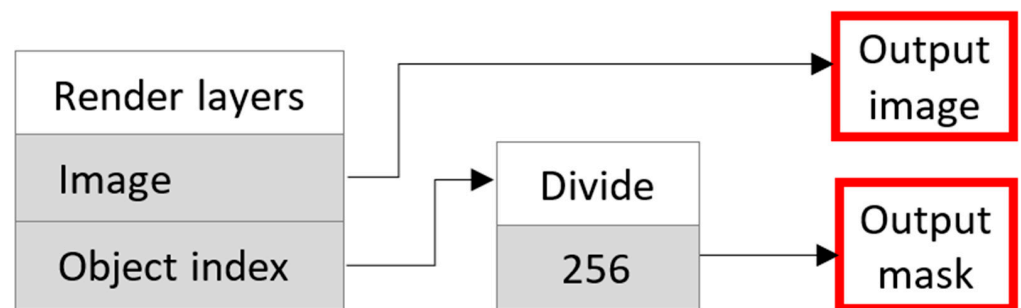


**Figure 7.** Detailed example of texture creation. (a) Single Principled BSDF node; (b) Principled BSDF node mixed with Glossy, Diffuse, and Transparent BSDFs; (c) combined BSDF material with Noise and Voronoi textures; (d) final output with added color ramp nodes to truncate the Bump heights and create transmission anisotropy in the Principled BSDF.

The developed shading node network does not reflect all possible and constantly expanding varieties of available 3D printing materials, but provides end users with an initial set of tools to change color, texture, and transparency parameters to achieve the required effects. The developed materials are available in the open-source file repository [56].

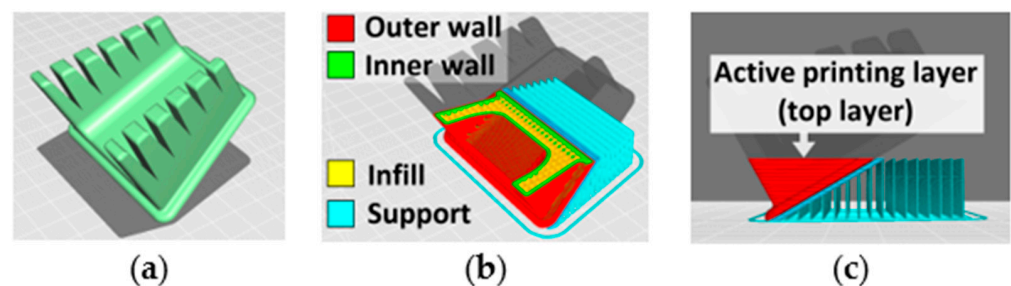
The G-code parsing procedure heavily utilizes the functionality of the Blender application programming interface [67], which provides access to the properties of all shader nodes used in the scene. The entire animation process is scripted with randomized locations of the camera, light sources, and printing bed/ground surface plane in timeline keyframes, while the graphics engine adds intermediate frames by interpolation. Most of the G-codes were used twice with different levels of part completion, material color, print surface texture, light source locations, and camera orientations.

The built-in compositing interface [68] was used to create pixel-perfect masks for each frame (Figure 8). During the slicing procedure, each extruder path acquires its own type, which can be visualized in pseudo colors in the slicing environment (Figure 9). In this work, the outer and inner walls were combined into one structural element “shell”. For visual segregation (masking) of individual scene elements (background, top layer, infill, shell, and support), different values of the object pass index parameter [69] were set at the G-code parsing stage. This allows each selected element to be rendered as a region filled with a certain grayscale level.



**Figure 8.** The composite node network (for internal structure segmentation, in this example) assigns user-defined color labels to each pixel in the output image, depending on whether it belongs to a particular area (infill, shell, or support) of the rendered part. This creates a pixel-precise ground truth mask (red) for each output image frame (red) in the animation.

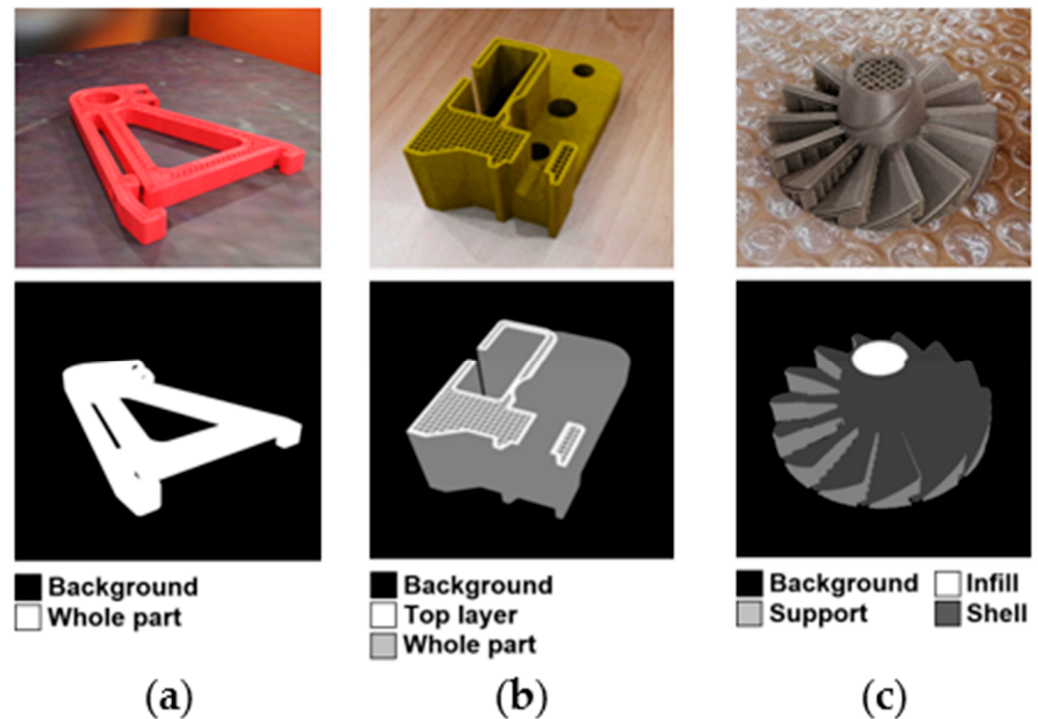
The internal physics-based path tracer Cycles [70] was used to render each frame of the animation. To reduce rendering time, the number of samples was set to 64, the total number of light path reflections was reduced to 8, and the Reflective and Refractive Caustics features were disabled. This rendering optimization may restrict the quality of the images produced but can greatly reduce the computational load. Cycles’ performance depends on the system’s computational power. An 8 GB GPU setup with a  $256 \times 256$  render tile size and an output image size of  $1024 \times 1024$  pixels takes up to one minute to process a single frame, depending on the scale and geometric complexity of the scene within the camera viewport. Rendering an entire 50-frame animation this way can take up to one hour.



**Figure 9.** 3D model slicing procedure. (a) Whole part in STL format. (b) Internal structure of sliced layers (red—outer shell, green—inner shell, yellow—infill, blue—support). (c) Side view illustrates current printing layer (top layer at each manufacturing stage).



(3) Synthetic image datasets: For the further task of semantic segmentation, three separate datasets were created (Figure 10). A total of 5763  $1024 \times 1024$  pixel image-mask pairs were generated for the segmentation of the entire 3D-printed part; 3570 for the top layer segmentation; and 1140 for the infill, shell, and support (internal layer structure) segmentation.



**Figure 10.** Image-mask pair samples for each AM synthetic dataset. (a) Whole part segmentation. (b) Top layer segmentation. (c) Internal layer segmentation.

### 3.2. Semantic Image Segmentation

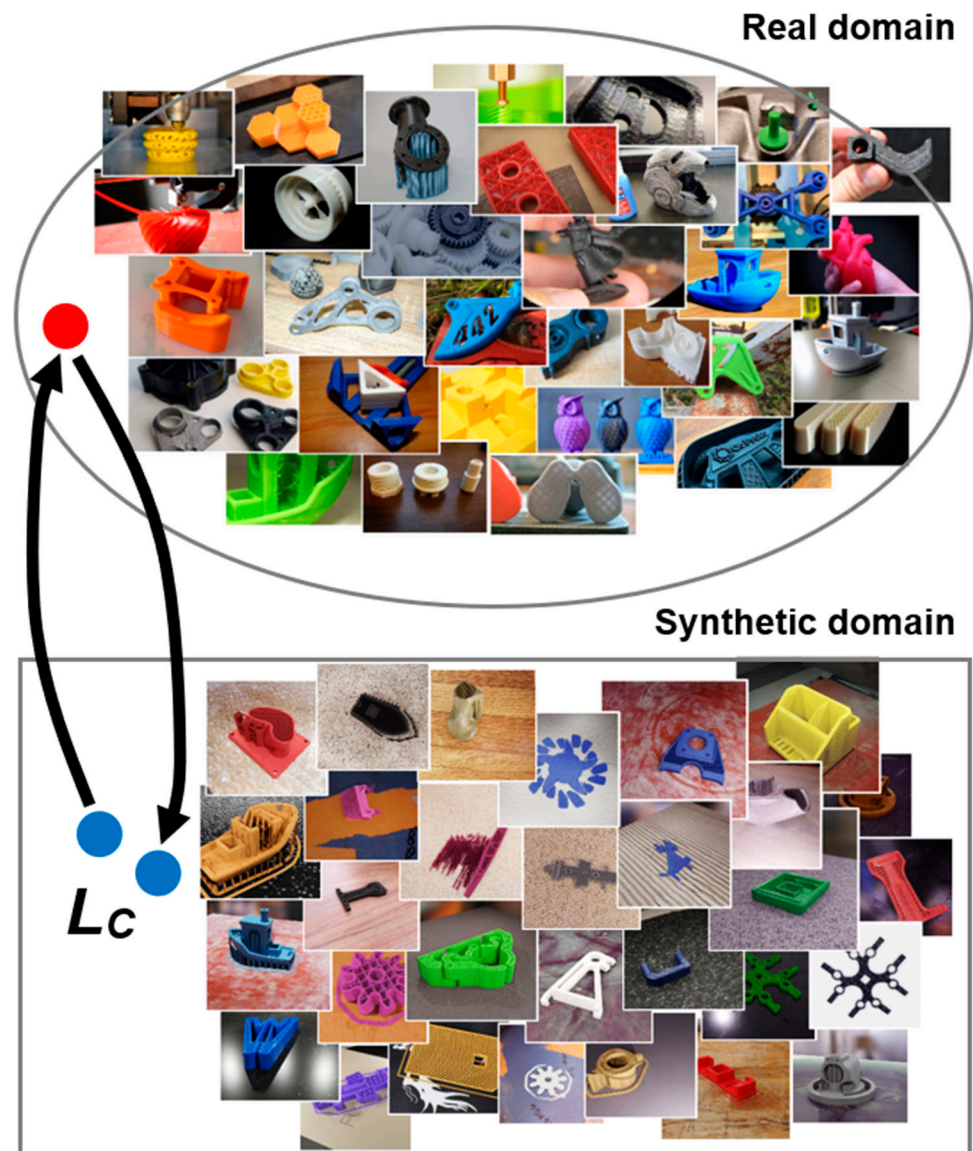
Minaee et al. [32], as well as Ulku and Akagunduz [40], presented a comprehensive overview of the modern research state in the field of semantic segmentation. As can be seen from the works [71–73], the U-Net family of neural network architectures has demonstrated high segmentation efficiency with small amounts of training data. The DeepLab architecture, in turn, is one of the basic architectures for subsequent domain adaptation [74–76].

This work employs the U-Net architecture [77] and its multi-class adaptation [78] due to its efficiency and simplicity.

### 3.3. Image-to-Image Translation

To potentially improve the efficiency of semantic segmentation, the application of the unpaired image-to-image translation method based on the CycleGAN network [39] was considered. The given method learns the mapping between the source domain (real images) and the target domain (synthetic images) by minimizing the cycle consistency loss  $L_C$  (Figure 11) in the absence of paired data samples.

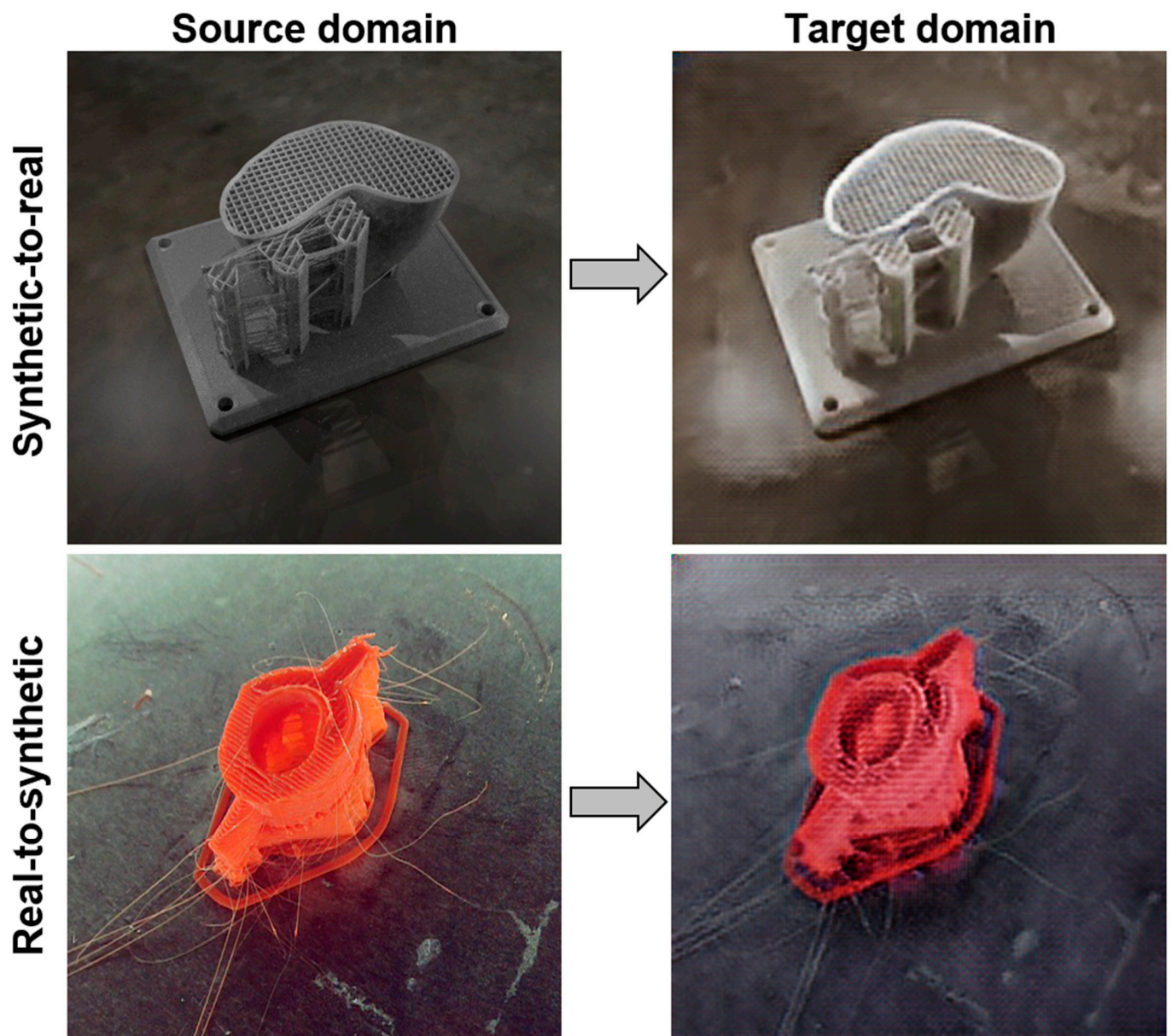




**Figure 11.** Unpaired image-to-image translation using the cycle-consistent adversarial network. Handpicked images of real and virtual printed parts were loaded into CycleGAN, which learns to map real domain images to their synthetic counterparts and vice versa, minimizing the cycle consistency loss  $L_c$ . Here, the red and blue circles represent the same image presented in different domains.

For this task, we manually selected 589 synthetic renders and 794 real images of 3D-printed parts. The learning result is two generators that convert the original images of the real domain into their synthetic counterparts, and vice versa (Figure 12).

As can be seen from Figure 12, translating a synthetic render into a real image makes colors more natural, while translating a real image into a synthetic one also reduces the contrast and saturation of both reflections on the printing bed/ground surface and incidental filament strings. This characteristic can improve segmentation in mediocre images.



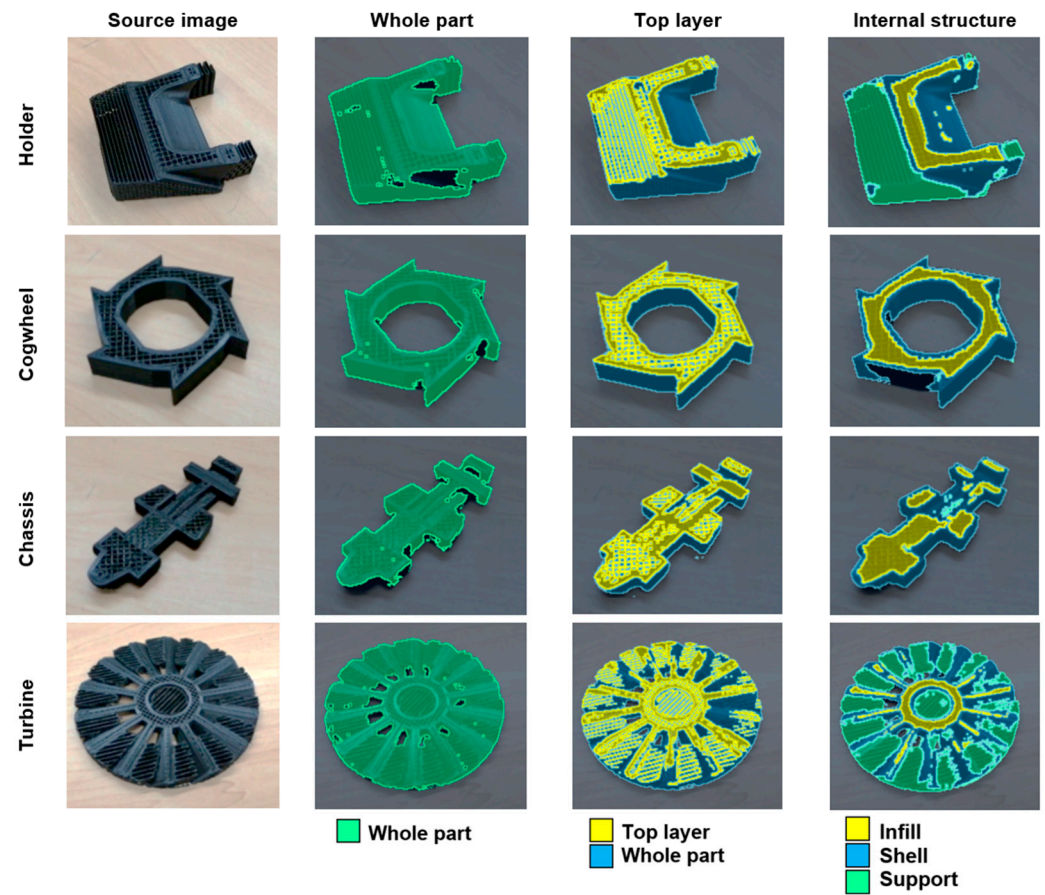
**Figure 12.** Image-to-image style translation example. Translating a real image into its synthetic version reduces the contrast and saturation of reflections and incidental filament strings.

#### 4. Results

The results of the semantic segmentation are presented using several real images, presented in Figure 13. The training of the neural network was carried out on synthetic renders without using the style translation technique.

Quantitative results are shown in Table 1. Test datasets include synthetic renders of STL models, both those included in the training dataset and those not included in it. The 3D models included in the training dataset have had their color, angle, and environmental parameters changed to avoid matching the data the model was trained on.





**Figure 13.** The results of semantic segmentation, presented using several real images. The neural network was trained on similar synthetic 3D models. The color, printing surface texture, and slicing parameters, however, differ from those used in the training dataset.

**Table 1.** Segmentation results for synthetic test datasets (mIoU scores, %).

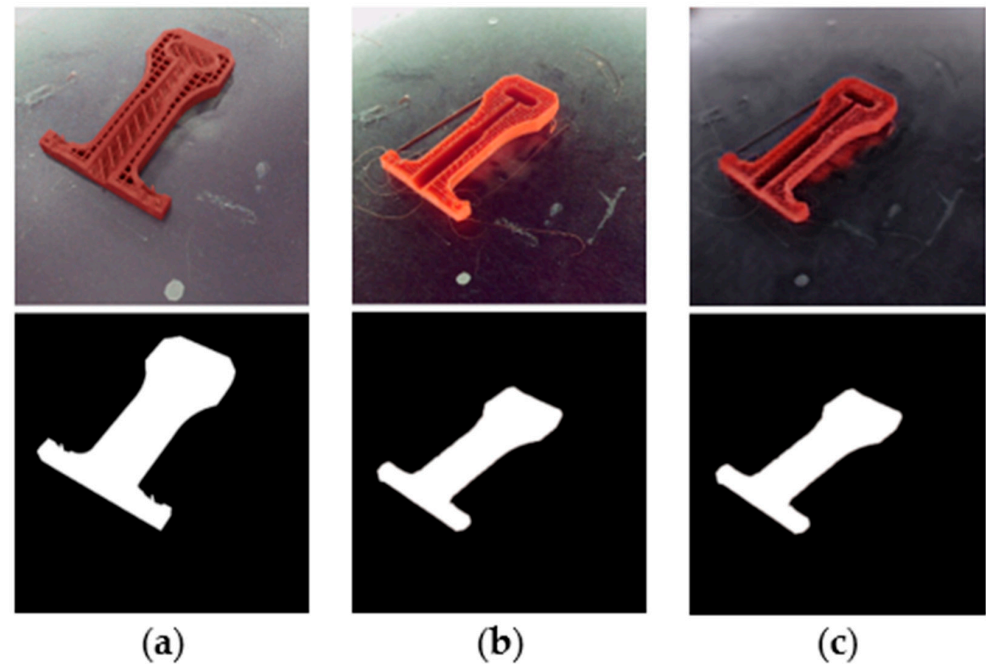
No. of Images	Test Dataset	Background	Top Layer	Shell	Support	Infill
89	Whole part segmentation (real images)	78.16	—	—	—	—
101	Whole part segmentation (synthetic render images)	94.90	—	—	—	—
68	Top layer segmentation (synthetic renders)	99.74	73.33	—	—	—
57	Internal structure segmentation (synthetic renders)	94.52	—	55.31	69.45	78.93

The intersection over union (IoU) quantifies the degree of overlap (from 0 to 100%) between the ground truth mask and the segmented pixel area of its predicted version, where a larger value indicates a more accurate segmentation, and the mIoU is the mean IoU value across the correspondent classes in the dataset. The calculation of mIoU scores for the real images was carried out only for the segmentation of the entire part, since obtaining manually-labeled ground truth masks for the top layer and the internal structure of the part is a nontrivial task, considering the geometric complexity of the filling elements.

As can be seen from Table 1, the segmentation accuracy on real images (78.16%) is inferior to that of the synthetic data (94.90%), which indicates the need for additional research on domain adaptation. Detecting the top layer is a more complex task for the neural network compared to segmenting the entire part, which is clearly noticeable in the results within the same dataset (mIoU 73.33% for the top layer, versus 99.74% for the background). Shell segmentation has the lowest score (mIoU 55.31%). This, apparently, is due to the variety of geometric shapes and the lack of a characteristic texture that the infill and support areas have. The segmentation efficiency of individual part elements depends

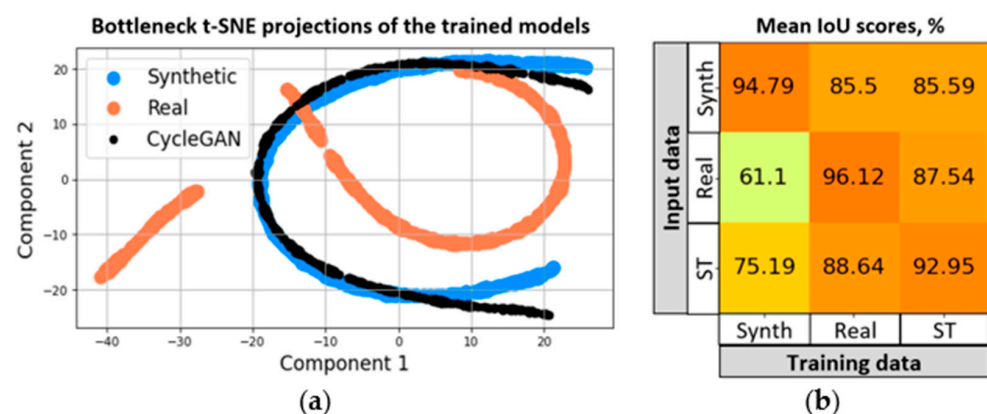
on their geometric complexity and the key factor for effective semantic segmentation is the number of image–mask pairs. With the developed open-source methodology, this database can be significantly expanded by end users, which will lead to increased segmentation accuracy among all available categories.

To analyze the influence of style transfer (ST) on semantic segmentation, separate CNN training of three datasets of one part was carried out (Figure 14). Synthetic and real datasets consist of 49 and 36 image–mask pairs, respectively.



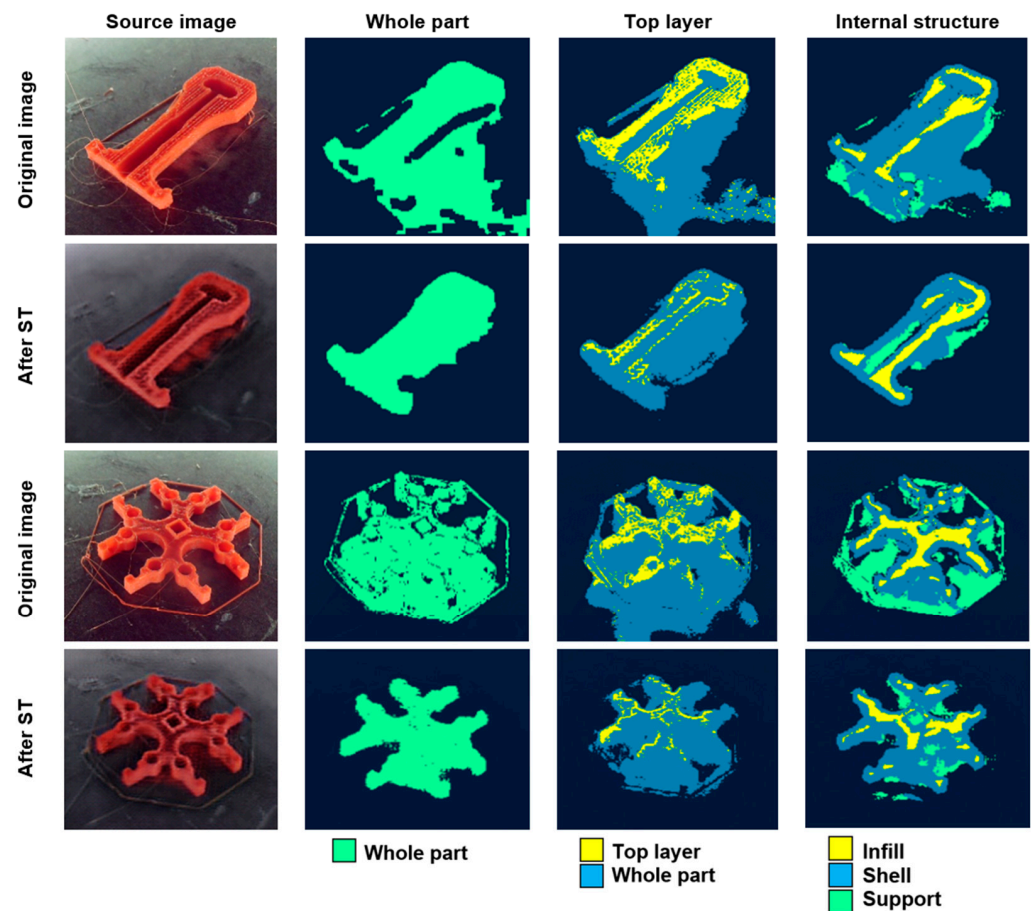
**Figure 14.** Datasets for the style transfer influence analysis. (a) Synthetic data. (b) Real data. (c) Real data after style transfer. The upper row shows sample images and the lower row illustrates the corresponding ground truth masks.

To compare the domains, we used t-distributed stochastic neighbor embedding (t-SNE) [79,80] projections of the normalized bottleneck layers of trained U-Net models (Figure 14). The nonlinear dimensionality reduction technique was applied to 512-dimensional normalized vectors in the narrowest parts of the trained models to visualize the affinity of the domains in latent feature space. As can be seen from Figure 15a, the feature space of the real domain (orange) is getting closer to synthetic data (blue) after the image-to-image style translation (black).



**Figure 15.** Domain comparison via t-SNE projections (a), and segmentation performance before and after style translation (b).

In addition to t-SNE projections, the segmentation performance of the real image data after ST was also analyzed (Figure 15b). The heatmap columns represent the data on which the neural network model was trained, and the rows stand for the input data to which segmentation was applied. The highest mIoU, as expected, was observed in those datasets on which the model was trained. When converting the real input data into ST using image-to-image translation, however, the segmentation score increased from 61.10% to 75.19% for the model trained solely on synthetic data. This parameter is the most valuable, since in real conditions, training a convolutional network on real data may not be possible due to the lack of ground truth masks. This indicates that the ST method as a precursor to domain adaptation can significantly improve real 3D printing image segmentation in situations where a model trained on synthetic data may be the only tool available. The sample results of image segmentation before and after style translation are shown in Figure 16.



**Figure 16.** The results of image segmentation before and after style translation. Real-to-synthetic style transfer reduces the saturation of the incidental filament strings and reflections on the printing platform, which, in turn, affects the results of semantic segmentation.

As can be seen from Figure 16, real-to-synthetic style transferring reduces the saturation of the incidental filament strings and reflections on the printing platform, which, in turn, affects the results of semantic segmentation. Image-to-image translation, therefore, could be a powerful tool in further improving segmentation performance through domain adaptation techniques.

This work continues the previous authors' research on the use of physical rendering and demonstrates the significant potential of using synthetic data and machine learning in the field of additive manufacturing. Due to the relative simplicity of virtual printing and training data generation, segmentation of the contours of a manufactured part can be performed at every stage of its creation using a single camera in an arbitrary position.



This reduces the requirements for camera calibration and eliminates the need to use visual markers to tightly bind the image frame to the coordinate system of the 3D printing space. It also offers the flexibility to be used on any type of 3D printing system with the addition of an after-market camera.

The limitations of the developed method are the need to create synthetic images and increase the training dataset for each new manufactured part, as well as the implementation of transfer learning to improve the segmentation accuracy. Additional research is also required in the field of domain adaptation applications based on existing state-of-the-art techniques [81–83].

Together with edge-based markerless tracking [84,85], the developed technique can become an integral part of a 3D printing control and monitoring system such as Octo-Print [86]. In the future, this will make it possible to implement an inline comprehensive system for recognizing the type of part being produced and determining its location and orientation in the workspace, as well as for tracking its manufacturing deviations.

## 5. Discussion and Conclusions

The proposed method is a novel approach to segmenting key regions of a part during its fabrication utilizing the information in the G-code and synthetic image data. The semantic segmentation framework for additive manufacturing can enhance the visual analysis of manufacturing processes and allow the detection of individual manufacturing errors, while significantly reducing the requirements for positioning accuracy and camera calibration.

The results of this work will allow the localization of 3D-printed parts in captured image frames, as well as the application of image processing techniques to the parts' structural elements to follow the tracking of manufacturing deviations. The use of image style transfer is of significant value for further research in the field of adapting the domain of synthetic renders to real images of 3D-printed products.

The methodology demonstrated achieved the following mIoU scores for the synthetic test datasets: entire printed part, 94.90%; top layer, 73.33%; infill, 78.93%; shell, 55.31%; support, 69.45%. Increasing the number of image–mask pairs used for training neural networks will improve segmentation accuracy. The results illustrate the effectiveness of the developed method, but also indicate the need for additional experiments to eliminate the synthetic-to-real domain gap.

This research presents ways to expand the number of segmentation categories by using the information in the G-code about the characteristics of separate extruded sections. Further study, however, is required to analyze the impacts of material and texture shaders, as well as lighting and rendering parameters, on segmentation efficiency.

Revealing this system to end users will allow constant expansion of the synthetic image database for subsequent neural network training and improvement of segmentation results. Integrating it with web-based 3D printing control systems can help to perform layer-wise analysis of manufactured parts, and also help to classify and track failures based on their bonding to a particular area of the model.

**Author Contributions:** Conceptualization, A.P. and J.M.P.; methodology, A.P. and J.M.P.; software, A.P., H.S. and H.D.; validation, A.P., H.S. and H.D.; formal analysis, A.P. and J.M.P.; resources, A.P. and J.M.P.; data curation, A.P., H.S. and H.D.; writing—original draft preparation, A.P., H.S., H.D. and J.M.P.; writing—review and editing, J.M.P. and A.P.; visualization, A.P., H.S. and H.D.; supervision, J.M.P.; project administration, J.M.P. and A.P.; funding acquisition, J.M.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Thompson Endowments and the Natural Sciences and Engineering Research Council of Canada.

**Data Availability Statement:** The data and source code for this project can be obtained from the Open Science Framework (OSF) repository, under J.M. Pearce, A. Petsiuk, Synthetic-to-real composite semantic segmentation in additive manufacturing; OSF Source file repository. Available at <https://osf.io/h8r45> (accessed on 27 March 2024).

**Acknowledgments:** The authors would like to thank Heinz Lopmeier for permission to use his code as a basis template for G-code parsing, as well as Doug Everett and Kenneth Jiang for access to Spaghetti Detective's user performance database.

**Conflicts of Interest:** The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

## References

1. Geyer, R.; Jambeck, J.R.; Law, K.L. Production, Use, and Fate of all Plastics Ever Made. *Sci. Adv.* **2017**, *3*, e1700782. [CrossRef] [PubMed]
2. Jambeck, J.R.; Geyer, R.; Wilcox, C.; Siegler, T.R.; Perryman, M.; Andrady, A.; Narayan, R.; Law, K.L. Plastic Waste Inputs From Land Into the Ocean. *Science* **2015**, *347*, 768–771. [CrossRef] [PubMed]
3. Laplume, A.O.; Petersen, B.; Pearce, J.M. Global value chains from a 3D printing perspective. *J. Int. Bus. Stud.* **2016**, *47*, 595–609. [CrossRef]
4. Petersen, E.E.; Pearce, J.M. Emergence of home manufacturing in the developed world: Return on investment for open-source 3-D printers. *Technologies* **2017**, *5*, 7. [CrossRef]
5. Pearce, J.M.; Qian, J.Y. Economic Impact of DIY Home Manufacturing of Consumer Products with Low-cost 3D Printing from Free and Open Source Designs. *Eur. J. Soc. Impact Circ. Econ.* **2022**, *3*, 1–24. [CrossRef]
6. Hunt, E.; Zhang, C.; Anzalone, N.; Pearce, J.M. Polymer recycling codes for distributed manufacturing with 3-D printers. *Resour. Conserv. Recycl.* **2015**, *97*, 24–30. [CrossRef]
7. Shahrubudin, N.; Lee, T.C.; Ramlan, R. An overview on 3D printing technology: Technological, materials, and applications. *Procedia Manuf.* **2019**, *35*, 1286–1296. [CrossRef]
8. Global 3D Printing Filament Market By Material, By Type, By End Use, By Region, Competition, Forecast & Opportunities, 2024. May, 2019. Available online: <https://www.reportbuyer.com/product/5778909/global-3d-printing-filament-market-by-material-by-typeby-end-use-by-regioncompetition-forecast-and-opportunities-2024.html> (accessed on 10 January 2024).
9. Wittbrodt, B.T.; Glover, A.G.; Laureto, J.; Anzalone, G.C.; Oppliger, D.; Irwin, J.L.; Pearce, J.M. Life-cycle economic analysis of distributed manufacturing with open-source 3-D printers. *Mechatronics* **2013**, *23*, 713–726. [CrossRef]
10. Sharp, S.; 3DQue Systems, Vancouver, BC, Canada. Personal communication, 4 June 2022.
11. Kang, H.D. Analysis of furniture design cases using 3D printing technique. *J. Korea Contents Assoc.* **2015**, *15*, 177–186. [CrossRef]
12. Bow, J.K.; Gallup, N.; Sadat, S.A.; Pearce, J.M. Open source surgical fracture table for digitally distributed manufacturing. *PLoS ONE* **2022**, *17*, e0270328. [CrossRef]
13. Novak, J.I.; O'Neill, J. A design for additive manufacturing case study: Fingerprint stool on a BigRep ONE. *Rapid Prototyp. J.* **2019**, *25*, 1069–1079. [CrossRef]
14. Petsiuk, A.; Lavu, B.; Dick, R.; Pearce, J.M. Waste Plastic Direct Extrusion Hangprinter. *Inventions* **2022**, *7*, 70. [CrossRef]
15. Woern, A.L.; Byard, D.J.; Oakley, R.B.; Fiedler, M.J.; Snabes, S.L.; Pearce, J.M. Fused particle fabrication 3-D printing: Recycled materials' optimization and mechanical properties. *Materials* **2018**, *11*, 1413. [CrossRef]
16. Oleff, A.; Kuster, B.; Stonis, M.; Overmeyer, L. Process monitoring for material extrusion additive manufacturing: A state-of-the-art review. *Prog. Addit. Manuf.* **2021**, *6*, 705–730. [CrossRef]
17. Ceruti, A.; Liverani, A.; Bombardi, T. Augmented vision and interactive monitoring in 3D printing process. *Int. J. Inter. Des. Manuf.* **2017**, *11*, 385–395. [CrossRef]
18. Nuchitprasitchai, S.; Roggemann, M.C.; Pearce, J.M. Factors effecting real-time optical monitoring of fused filament 3D printing. *Prog. Addit. Manuf. J.* **2017**, *2*, 133–149. [CrossRef]
19. Johnson, A.; Zarezadeh, H.; Han, X.; Bibb, R.; Harris, R. Establishing in-process inspection requirements for material extrusion additive manufacturing. In Proceedings of the Fraunhofer Direct Digital Manufacturing Conference, Berlin, Germany, 16–17 March 2016.
20. Hurd, S.; Camp, C.; White, J. Quality assurance in additive manufacturing through mobile computing. In *Mobile Computing, Applications, and Services: 7th International Conference, MobiCASE 2015, Berlin, Germany, 12–13 November 2015*; Springer: Cham, Switzerland, 2015; pp. 203–220.
21. Jeong, H.; Kim, M.; Park, B.; Lee, S. Vision-Based Real-Time Layer Error Quantification for Additive Manufacturing. In Proceedings of the International Manufacturing Science And Engineering Conference, Los Angeles, CA, USA, 4 June 2017.
22. Wasserfall, F.; Ahlers, D.; Hendrich, N. Optical In-Situ Verification of 3D-Printed Electronic Circuits. In Proceedings of the 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, 22–26 August 2019; pp. 1302–1307. [CrossRef]
23. Straub, J. 3D printing cybersecurity: Detecting and preventing attacks that seek to weaken a printed object by changing fill level. In Proceedings of the Dimensional Optical Metrology and Inspection for Practical Applications VI, Anaheim, CA, USA, 9 June 2017. [CrossRef]

24. Kutzer, M.D.; DeVries, L.D.; Blas, C.D. Part monitoring and quality assessment of conformal additive manufacturing using image reconstruction. In Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Quebec City, QC, Canada, 26–29 August 2018. [CrossRef]
25. Chen, Z.; Horowitz, R. Vision-assisted Arm Motion Planning for Freeform 3D Printing. In Proceedings of the 2019 American Control Conference (ACC), Philadelphia, PA, USA, 10–12 July 2019; pp. 4204–4209. [CrossRef]
26. Shen, H.; Du, W.; Sun, W.; Xu, Y.; Fu, J. Visual detection of surface defects based on self-feature comparison in robot 3-D printing. *Appl. Sci.* **2020**, *10*, 235. [CrossRef]
27. Malik, A.; Lhachemi, H.; Ploennigs, J.; Ba, A.; Shorten, R. An application of 3D model reconstruction and augmented reality for real-time monitoring of additive manufacturing. *Procedia CIRP* **2019**, *81*, 346–351. [CrossRef]
28. Petsiuk, A.; Pearce, J.M. Open source computer vision-based layerwise 3D printing analysis. *Addit. Manuf.* **2020**, *36*, 101473. [CrossRef]
29. Petsiuk, A.; Pearce, J.M. Towards smart monitored AM: Open source in-situ layer-wise 3D printing image anomaly detection using histograms of oriented gradients and a physics-based rendering engine. *Addit. Manuf.* **2022**, *52*, 102690. [CrossRef]
30. Spaghetti Detective. Available online: <https://www.obico.io/the-spaghettidetector.html> (accessed on 10 January 2024).
31. The Spaghetti Detective Plugin. Available online: <https://github.com/TheSpaghettiDetective/OctoPrintTheSpaghettiDetective> (accessed on 10 January 2024).
32. Minaee, S.; Boykov, Y.; Porikli, F.; Plaza, A.; Kehtarnavaz, N.; Terzopoulos, D. Image Segmentation Using Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 3523–3542. [CrossRef] [PubMed]
33. Blender: The Free and Open Source 3D Creation Suite. Available online: <https://www.blender.org> (accessed on 10 January 2024).
34. Csurka, G.; Volpi, R.; Chidlovskii, B. Unsupervised Domain Adaptation for Semantic Image Segmentation: A Comprehensive Survey. *arXiv* **2021**, arXiv:2112.03241.
35. Farahani, A.; Voghoei, S.; Rasheed, K.; Arabnia, H.R. A Brief Review of Domain Adaptation. In *Advances in Data Science and Information Engineering. Transactions on Computational Science and Computational Intelligence*; Springer: Cham, Switzerland, 2021.
36. Imbusch, B.; Schwarz, M.; Behnke, S. Synthetic-to-Real Domain Adaptation using Contrastive Unpaired Translation. *arXiv* **2022**, arXiv:2203.09454.
37. Li, P.; Liang, X.; Jia, D.; Xing, E.P. Semantic-aware Grad-GAN for Virtual-to-Real Urban Scene Adaption. *arXiv* **2018**, arXiv:1801.01726.
38. Lee, S.; Park, E.; Yi, H.; Lee, S.H. StRDAN: Synthetic-to-Real Domain Adaptation Network for Vehicle Re-Identification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Seattle, WA, USA, 13–19 June 2020.
39. Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2242–2251. [CrossRef]
40. Ulku, I.; Akagund, E. A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D Images. *Appl. Artif. Intell.* **2022**, *36*, 2032924. [CrossRef]
41. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.
42. Richter, S.R.; Vineet, V.; Roth, S.; Koltun, V. Playing for Data: Ground Truth from Computer Games. *arXiv* **2016**, arXiv:1608.02192.
43. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 3234–3243. [CrossRef]
44. Nikolenko, S.I. *Synthetic Data for Deep Learning*. SOIA; Springer: Cham, Switzerland, 2021; Volume 174. [CrossRef]
45. de Melo, C.M.; Torralba, A.; Guibas, L.; DiCarlo, J.; Chellappa, R.; Hodgins, J. Next-generation deep learning based on simulators and synthetic data. *Trends Cogn. Sci.* **2022**, *26*, 174–187. [CrossRef] [PubMed]
46. Ward, D.; Moghadam, P.; Hudson, N. Deep Leaf Segmentation Using Synthetic Data. *arXiv* **2018**, arXiv:1807.10931.
47. Boikov, A.; Payor, V.; Savelev, R.; Kolesnikov, A. Synthetic Data Generation for Steel Defect Detection and Classification Using Deep Learning. *Symmetry* **2021**, *13*, 1176. [CrossRef]
48. Valizadeh, M.; Wolff, S.J. Convolutional Neural Network applications in additive manufacturing: A review. *Adv. Ind. Manuf. Eng.* **2022**, *4*, 100072. [CrossRef]
49. Banadaki, Y.; Razaviarab, N.; Fekrmandi, H.; Sharifi, S. Toward Enabling a Reliable Quality Monitoring System for Additive Manufacturing Process using Deep Convolutional Neural Networks. *arXiv* **2020**, arXiv:2003.08749.
50. Saluja, A.; Xie, J.; Fayazbakhsh, K. A closed-loop in-process warping detection system for fused filament fabrication using convolutional neural networks. *J. Manuf. Proc.* **2020**, *58*, 407–415. [CrossRef]
51. Jin, Z.; Zhang, Z.; Gu, G.X. Automated Real-Time Detection and Prediction of Interlayer Imperfections in Additive Manufacturing Processes Using Artificial Intelligence. *Adv. Intell. Syst.* **2019**, *2*, 1900130. [CrossRef]
52. Brion, D.A.J.; Pattinson, S.W. Generalisable 3D printing error detection and correction via multi-head neural networks. *Nat. Commun.* **2022**, *13*, 4654. [CrossRef]

53. Wong, V.W.H.; Ferguson, M.; Law, K.H.; Lee, Y.T.; Witherell, P. Segmentation of Additive Manufacturing Defects Using U-Net. *ASME J. Comput. Inf. Sci. Eng.* **2022**, *22*, 031005. [CrossRef]
54. Cannizzaro, D.; Varrella, A.G.; Paradiso, S.; Sampieri, R.; Chen, Y.; Macii, A.; Patti, E.; Di Cataldo, S. In-Situ Defect Detection of Metal Additive Manufacturing: An Integrated Framework. *IEEE Trans. Emerg. Top. Comput.* **2022**, *10*, 74–86. [CrossRef]
55. Daytalab, O.; Kazemian, A.; Yuan, X.; Khoshnevis, B. Automated inspection in robotic additive manufacturing using deep learning for layer deformation detection. *J. Intell. Manuf.* **2022**, *33*, 771–784. [CrossRef]
56. Pearce, J.M.; Petsiuk, A. Synthetic-to-Real Composite Semantic Segmentation in Additive Manufacturing. OSF Source File Repository. Available online: <https://osf.io/h8r45> (accessed on 10 January 2024).
57. Thingiverse: An Open Catalog of Computer-Aided Designs for 3D Printing. Available online: <https://www.thingiverse.com> (accessed on 10 January 2024).
58. MatterControl: 3D Printing Software. Available online: <https://www.matterhackers.com/store/1/mattercontrol/sk/MKZGTDW6> (accessed on 10 January 2024).
59. Lopmeier, H. Blender-Gcode-Importer. Available online: <https://github.com/Heinz-Loepmeier/Blender-Gcode-Import> (accessed on 10 January 2024).
60. Blender: Shader Nodes Library. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/index.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/index.html) (accessed on 10 January 2024).
61. Blender: Noise Texture Node. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/textures/noise.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/textures/noise.html) (accessed on 10 January 2024).
62. Blender: Voronoi Texture Node. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/textures/voronoi.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/textures/voronoi.html) (accessed on 10 January 2024).
63. Blender: Principled BSDF. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/shader/principled.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/shader/principled.html) (accessed on 10 January 2024).
64. Blender: Glossy BSDF. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/shader/glossy.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/shader/glossy.html) (accessed on 10 January 2024).
65. Blender: Diffuse BSDF. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/shader/diffuse.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/shader/diffuse.html) (accessed on 10 January 2024).
66. Blender: Transparent BSDF. Available online: [https://docs.blender.org/manual/en/3.4/render/shader\\_nodes/shader/transparent.html](https://docs.blender.org/manual/en/3.4/render/shader_nodes/shader/transparent.html) (accessed on 10 January 2024).
67. Blender API. Available online: <https://docs.blender.org/api/current/index.html> (accessed on 10 January 2024).
68. Blender Compositing. Available online: <https://docs.blender.org/manual/en/3.4/compositing/index.html> (accessed on 10 January 2024).
69. Blender: Object Pass Index. Available online: <https://docs.blender.org/manual/en/3.4/render/layers/passes.html> (accessed on 10 January 2024).
70. Blender Cycles. Available online: <https://docs.blender.org/manual/en/3.4/render/cycles/index.html> (accessed on 10 January 2024).
71. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597.
72. Qin, X.; Zhang, Z.; Huang, C.; Dehghan, M.; Zaiane, O.R.; Jagersand, M. U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection. *arXiv* **2020**, arXiv:2005.09007. [CrossRef]
73. Huang, H.; Lin, L.; Tong, R.; Hu, H.; Zhang, Q.; Iwamoto, Y.; Han, X.; Chen, Y.-W.; Wu, J. UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation. *arXiv* **2020**, arXiv:2004.08790.
74. Toldo, M.; Michieli, U.; Zanuttigh, P. Unsupervised Domain Adaptation in Semantic Segmentation via Orthogonal and Clustered Embeddings. *arXiv* **2020**, arXiv:2011.12616.
75. Yang, J.; Li, C.; An, W.; Ma, H.; Guo, Y.; Rong, Y.; Zhao, P.; Huang, J. Exploring Robustness of Unsupervised Domain Adaptation in Semantic Segmentation. *arXiv* **2021**, arXiv:2105.10843.
76. Guo, X.; Yang, C.; Li, B.; Yuan, Y. MetaCorrection: Domain-aware Meta Loss Correction for Unsupervised Domain Adaptation in Semantic Segmentation. *arXiv* **2021**, arXiv:2103.05254.
77. Buda, M. U-Net for Brain Segmentation. 2019. Available online: <https://pytorch.org/hub/mateuszbudabrain-segmentation-pytorchUNET> (accessed on 10 January 2024).
78. Battocchio, F. U-Net Architecture for Multiclass Semantic Segmentation. 2020. Available online: <https://github.com/France1/unet-multiclasspytorch> (accessed on 10 January 2024).
79. Hinton, G.E.; Roweis, S.T. Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems*; Becker, S., Thrun, S., Obermayer, K., Eds.; MIT Press: Cambridge, CA, USA, 2002; Volume 15.
80. van der Maaten, L.J.P.; Hinton, G.E. Visualizing High-Dimensional Data Using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
81. Xu, T.; Chen, W.; Wang, P.; Wang, F.; Li, H.; Jin, R. CDTrans: Cross-domain Transformer for Unsupervised Domain Adaptation. *arXiv* **2022**, arXiv:2109.06165.
82. Xie, B.; Li, S.; Li, M.; Liu, C.H.; Huang, G.; Wang, G. SePiCo: SemanticGuided Pixel Contrast for Domain Adaptive Semantic Segmentation. *arXiv* **2022**, arXiv:2204.08808. [CrossRef] [PubMed]

- 
83. Hoyer, L.; Dai, D.; Van Gool, L. HRDA: Context-Aware High-Resolution Domain-Adaptive Semantic Segmentation. *arXiv* **2022**, arXiv:2204.13132.
  84. Han, P.; Zhao, G. A review of edge-based 3D tracking of rigid objects. *Virtual Real. Intell. Hardw.* **2019**, *1*, 580–596. [[CrossRef](#)]
  85. Wang, B.; Zhong, F.; Qin, X. Robust edge-based 3D object tracking with direction-based pose validation. *Multimed. Tools Appl.* **2019**, *78*, 12307–12331. [[CrossRef](#)]
  86. OctoPrint: An Open Source 3D Printer Controller Application. Available online: <https://octoprint.org> (accessed on 10 January 2024).

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.