Machine learning modeling for image segmentation in manufacturing and agriculture applications

by

Mohammad Najjartabar Bisheh

B.S., Mazandaran University of Science and Technology, 2012
M.S., Amirkabir University of Technology, 2015

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Industrial and Manufacturing Systems Engineering
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Abstract

This dissertation focuses on applying machine learning (ML) modeling for image segmentation tasks of various applications such as additive manufacturing monitoring, agricultural soil cover classification, and laser scribing quality control. The proposed framework uses various ML models such as gradient boosting classifier and deep convolutional neural network to improve and automate image segmentation tasks.

In recent years, supervised ML methods have been widely adopted for imaging processing applications in various industries. The presence of cameras installed in production processes has generated a vast amount of image data that can potentially be used for process monitoring. Specifically, deep supervised machine learning models have been successfully implemented to build automatic tools for filtering and classifying useful information for process monitoring. However, successful implementations of deep supervised learning algorithms depend on several factors such as distribution and size of training data, selected ML models, and consistency in the target domain distribution that may change based on different environmental conditions over time.

The proposed framework takes advantage of general-purposed, trained supervised learning models and applies them for process monitoring applications related to manufacturing and agriculture. In Chapter 2, a layer-wise framework is proposed to monitor the quality of 3D printing parts based on top-view images. The proposed statistical process monitoring method starts with self-start control charts that require only two successful initial prints. Unsupervised machine learning methods can be used for problems in which high accuracy is not required, but statistical process monitoring usually demands high classification accuracies to avoid Type I and II errors. Answering the challenges of image processing using unsupervised methods due to lighting, a

supervised Gradient Boosting Classifier (GBC) with 93 percent accuracy is adopted to classify each printed layer from the printing bed.

Despite the power of GBC or other decision-tree-based ML models comparable to unsupervised ML models, their capability is limited in terms of accuracy and running time for complex classification problems such as soil cover classification. In Chapter 3, a deep convolutional neural network (DCNN) for semantic segmentation is trained to quantify and monitor soil coverage in agricultural fields. The trained model is capable of accurately quantifying green canopy cover, counting plants, and classifying stubble. Due to the wide variety of scenarios in a real agricultural field, 3942 high-resolution images were collected and labeled for training and test data set.

The difficulty and hardship of collecting, cleaning, and labeling the mentioned dataset was the motivation to find a better approach to alleviate data-wrangling burden for any ML model training. One of the most influential factors is the need for a high volume of labeled data from an exact problem domain in terms of feature space and distributions of data of all classes. Image data preparation for deep learning model training is expensive in terms of the time for labelling due to tedious manual processing. Multiple human labelers can work simultaneously but inconsistent labeling will generate a training data set that often compromises model performance. In addition, training a ML model for a complication problem from scratch will also demand vast computational power.

One of the potential approaches for alleviating data wrangling challenges is transfer learning (TL). In Chapter 4, a TL approach was adopted for monitoring three laser scribing characteristics – scribe width, straightness, and debris to answer these challenges. The proposed transfer deep convolutional neural network (TDCNN) model can reduce timely and costly processing of data preparation. The proposed framework leverages a deep learning model already trained for a similar problem and only uses 21 images generated gleaned from the problem domain. The proposed TDCNN overcame the data challenge by leveraging the DCNN model called VGG16 already trained for basic geometric features using more than two million pictures. Appropriate image processing techniques were provided to measure scribe width and line straightness as well as total scribe and debris area using classified images with 96 percent accuracy. In addition to the fact that the TDCNN is functioning with less trainable parameters (i.e., 5 million versus 15 million for VGG16), increasing training size to 154 did not provide significant improvement in accuracy that shows the TDCNN does not need high volume of data to be successful. Finally, chapter 5 summarizes the proposed work and lays out the topics for future research.

Machine learning modeling for image segmentation in manufacturing and agriculture applications

by

Mohammad Najjartabar Bisheh

B.S., Mazandaran University of Science and Technology, 2012
M.S., Amirkabir University of Technology, 2015

A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Industrial and Manufacturing Systems Engineering
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Approved by:

Major Professor
Shing I. Chang

Copyright

Abstract

This dissertation focuses on applying machine learning (ML) modelling for image segmentation tasks of various applications such as additive manufacturing monitoring, agricultural soil cover classification, and laser scribing quality control. The proposed ML framework uses various ML models such as gradient boosting classifier and deep convolutional neural network to improve and automate image segmentation tasks.

In recent years, supervised ML methods have been widely adopted for imaging processing applications in various industries. The presence of cameras installed in production processes has generated a vast amount of image data that can potentially be used for process monitoring. Specifically, deep supervised machine learning models have been successfully implemented to build automatic tools for filtering and classifying useful information for process monitoring. However, successful implementations of deep supervised learning algorithms depend on several factors such as distribution and size of training data, selected ML models, and consistency in the target domain distribution that may change based on different environmental conditions over time.

The proposed framework takes advantage of general-purposed, trained supervised learning models and applies them for process monitoring applications related to manufacturing and agriculture. In Chapter 2, a layer-wise framework is proposed to monitor the quality of 3D printing parts based on top-view images. The proposed statistical process monitoring method starts with self-start control charts that require only two successful initial prints. Unsupervised machine learning methods can be used for problems in which high accuracy is not required, but statistical process monitoring usually demands high classification accuracies to avoid Type I and II errors. Answering the challenges of image processing using unsupervised methods due to lighting, a

supervised Gradient Boosting Classifier (GBC) with 93 percent accuracy is adopted to classify each printed layer from the printing bed.

Despite the power of GBC or other decision-tree-based ML models to comparable to unsupervised ML models, their capability is limited in terms of accuracy and running time for complex classification problems such as soil cover classification. In Chapter 3, a deep convolutional neural network (DCNN) for semantic segmentation is trained to quantify and monitor soil coverage in agricultural fields. The trained model is capable of accurately quantifying green canopy cover, counting plants, and classifying stubble. Due to the wide variety of scenarios in a real agricultural field, 3942 high-resolution images were collected and labeled for training and test data set.

The difficulty and hardship of collecting, cleaning, and labeling the mentioned dataset was the motivation to find a better approach to alleviate data-wrangling burden for any ML model training. One of the most influential factors is the need for a high volume of labeled data from an exact problem domain in terms of feature space and distributions of data of all classes. Image data preparation for deep learning model training is expensive in terms of the time for labelling due to tedious manual processing. Multiple human labelers can work simultaneously but inconsistent labeling will generate a training data set that often compromises model performance. In addition, training a ML model for a complication problem from scratch will also demand vast computational power.

One of the potential approaches for alleviating data wrangling challenges is transfer learning (TL). In Chapter 4, a TL approach was adopted for monitoring three laser scribing characteristics – scribe width, straightness, and debris to answer these challenges. The proposed transfer deep convolutional neural network (TDCNN) model can reduce timely and costly processing of data preparation. The proposed framework leverages a deep learning model already trained for a similar problem and only uses 21 images generated gleaned from the problem domain. The proposed TDCNN overcame the data challenge by leveraging the DCNN model called VGG16 already trained for basic geometric features using more than two million pictures. Appropriate image processing techniques were provided to measure scribe width and line straightness as well as total scribe and debris area using classified images with 96 percent accuracy. In addition to the fact that the TDCNN is functioning with less trainable parameters (i.e., 5 million versus 15 million for VGG16), increasing training size to 154 did not provide significant improvement in accuracy that shows the TDCNN does not need high volume of data to be successful. Finally, chapter 5 summarizes the proposed work and lays out the topics for future research.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would love to express my sincere gratitude to professors, graduate students, and researchers who helped me during my Ph.D. period. First, thanks to my major advisor, Prof. Shing Chang, who was always there to help me advancing my research and studies. My special thanks to Dr. Patrignani who spent all the time I wanted for me and taught me lessons that I have never learned anywhere else.

I would love to thank to Prof. Lei and Dr. Sinha for their kind advice and help in my research and my life. Also, thanks to the department of Industrial and Manufacturing Systems Engineering at Kansas State University for giving me the opportunity to be a teacher assistant, teacher, and a Ph.D. researcher with all support and love. Thanks to all my friend in Manhattan who made Kansas home for me and became my family. Kansas will be a home for me and my friends in Manhattan will remain my family in my heart forever.

Finally, my sincere thanks and appreciation to my parents, my brother, and my sister for years of unconditional love and support.

Dedication

I dedicate this dissertation to my parents who made this possible with all support and scarifies.

1. Introduction

Manufacturing processes have been integrating with industrial digital technologies such as Artificial Intelligence (AI), Internet of Things (IoT), and cloud computing in the fourth industrial revolution [1]. Sensors and cameras installed on production machines and vehicles, satellites, drones, and many other data collection tools collect real-time data that could be potentially used for agriculture and manufacturing applications. A significant portion of collected data is either in the form of grayscale or RGB images. Image processing techniques for pre-processing, processing and post-processing are often necessary to extract useful information before data can be fed into algorithms or models [2]. Pre-processing operations include image cleaning, noise reductions, and labeling. Typical noise types include Gaussian noise, salt-and-pepper noise, and shot noise. Next, pre-processed images are processed further to extract useful information such as labeling specific objects with their location, width, height, etc. At the end, the extracted information must be validated in post processing.

1.1. Motivation and background

1.1.1. Image-based process monitoring using ML

Image processing consists of methods and operations on a set of gray scale or color arrays representing an image to extract some useful information [2]. With the advent of machine vision and ML, opportunities arise for the use of AI framework for monitoring and characterizing manufacturing processes as well as natural resource management. Image-based process monitoring has been used in a wide range of industries from agriculture and food production [3], [4] to health system monitoring [5]–[7], and manufacturing processes [8], [9]. Among all state-of-the-art monitoring methods, data-driven methods without physical models or expert knowledge have won massive popularity in recent years [10]. A wide variety of quality characteristics such as product geometry, surface analysis, dimensional data, and defect patterns can be monitored using imaged based monitoring [11].

Image-based process monitoring includes the following steps: image acquisition, image pre-processing, feature extraction, and process monitoring, and control charts [12]. Previous research in this area mostly focused on two approaches [13]. The first approach uses spatial control chart to detect the location of and size of a defect [14]–[18] while the second approach aims to detect defects through a well-defined statistical method [19], [20]. These two approaches may be implemented simultaneously [21]. Image-based process monitoring is discussed with more details in section 1.1.2. Although these advancements have brought considerable benefits, such as lower costs, production efficiency and consistency in monitoring, the potential of using generated images for process monitoring has not been discovered completely. Image-based process monitoring could also be used in different industries for proactive maintenance, packaging inspections, assembly line inspections, quality monitoring, tracking, and tracing, and demand prediction. Image-based process monitoring could be implemented via various ML techniques including supervised, unsupervised, and reinforcement learning. In the following sections, the basic definitions of ML techniques and their relations are described to outline the scope of this dissertation.

1.1.1.1. Unsupervised machine learning

Let's have domain D = $\{x_1, x_2, x_3, \dots, x_M\} \in \chi$ where $x_i \in R^n$ is input training instances with i.i.d. distribution. In unsupervised ML, the goal is to learn the internal structure of the data without knowing the labels. Dimensionality reduction methods (e.g., principal component analysis (PCA)), clustering (e.g., k-means), and density estimation are some of most common unsupervised methods.

1.1.1.2. Supervised machine learning

In supervised ML, label or labels are associated to each instance. Let's have domain D = $\{(x_1,y_1)$ $(x_2,y_2), \dots, (x_M,y_M)\} \sqsubseteq \chi \times \mathcal{Y}$ where $x_i \in R^n$ is the input instance, and $y_i$is its label. $\mathcal{Y}$ is called label space and χ is feature space or input space. The main goal in supervised learning is to learn a function h such that h(χ) is an estimator for $\mathcal{Y}$ with a high probability. We use GBC and neural networks (NN) in this dissertation. Other supervised ML methods include naïve bayes, logistics regression, support vector machine (SVM), and random forest are some of the popular supervised learning methods.

1.1.1.3. Reinforcement learning

Reinforcement learning (RL) is the task of learning through trial and error and rewarding desired behaviors or punishing undesired ones. RL is out of the scope of this research, but to understand the contribution of this research we needed to define it. RL could be supervised or unsupervised, where the difference between supervised learning and RL is that the feedback provided to the agent in supervised learning is a correct set of action for performing a task, but RL uses rewards policy. To compare to unsupervised learning, the goal in unsupervised learning is to find similarities, however, the goal in RL is to maximize rewards.

1.1.1.4. Deep learning

Deep learning (DL) is a powerful approach that was explicitly addressed first time in 1976 [22] but has gotten a momentum in applications and improvements of ML models recently. DL imitates the way how a human brain learns certain types of knowledge through multi-layered non-linear function approximators, typically neural networks. It should be noted that DL is not an ML model itself, but instead, it is a toolbox that can be applied to all other ML models. That is why DL could have overlapped with all other ML methods shown in Figure 1.1.

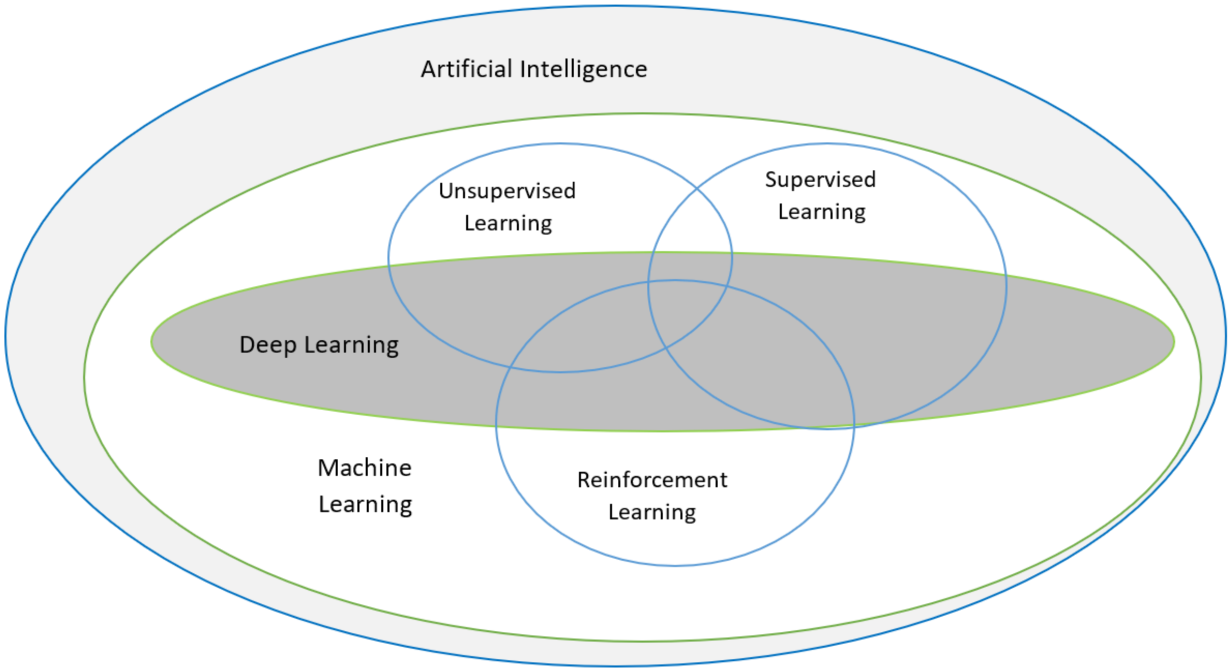


Figure 1.1. Relationship among AI, ML, supervised learning, unsupervised learning, DL, and RL

1.1.2. Process monitoring using transfer learning

Many challenges still exist in adopting DL framework for process monitoring. Large data requirements, the inability to produce physically consistent results, and their lack of generalizability to out-of-sample scenarios are some of the most important constraints in the application of black-box ML models, that causes limited success in scientific domains [23]. DL and convolutional neural network (CNN) have showed promising performance among other ML methods in recent years in different contexts from autonomous driving vehicles to medical image analysis [24], [25]. However, these methods need a significant amount of data for training a model with adequate accuracy [26]–[28]. Collecting image data may not be a big challenge these days but pre-processing and labeling of these images for training is. This data preparation stage is the most time-consuming and costly step in any machine vision/DL application. One way to alleviate this problem is the use of transfer learning [29].

Existing supervised ML methods such as decision tree and other methods based on various trees such as Random Forest (RF) or GBC may not need as much training data as the DL/CNN models do but still require a large amount of data. In addition, these traditional ML methods cannot handle complicated problems such as semantic segmentation with multiple quality characteristics. For example, there might be several classes of objects to be classified or each class might have different geometrical shapes and colors [28].

Transfer Learning (TL) may be used to alleviate the lack of labeled data problems. Specifically, TL enables model knowledge gained from a different yet similar problem for solving another problem. For the diagnostic radiology example, labelled images are often hard to obtain, especially for rare conditions. Another often-happened issue in ML training is overfitting which prevents a trained model from predicting instances accurately when they are not a part of the training dataset. Therefore, TL provides a mean for model training even when the training data set is relatively small. Usually, TL is best applied where the base model has been trained on a vastly bigger training set than the task at hand. Ng (2016) predicted that TL will be the next driver of ML commercial success after supervised learning [30]. Figure 1.2 shows a general structure of TL using a feature extraction approach.

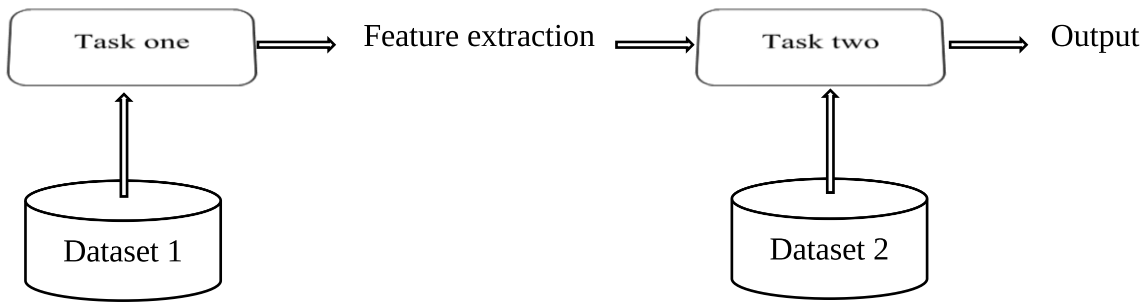


Figure 1.2. General structure of feature- representation – transfer approach

Figure 1.2 shows a general structure of TL using a feature extraction approach. Feature extraction is the process of reducing the dimension of a problem in which the raw data matrix is reduced to a smaller manageable groups [31]. TL can be defined in terms of domain, task, and marginal probabilities [32]. Let's say X= {x_1, x_2, …. , x_n} which x_i is a specific sample point and X ∈ χ. Domain D is defined as a tuple of feature space χ (e.g. Dataset 1 in Figure 1.2) and marginal probability P(X). A task, T, on the other hand, can be defined as a two-element tuple of the label space γ (e.g. Dataset 2 in Figure 1.2). The task is to maximize objective function f given the sample point X [33].

$$D = \{ \chi , P(X)\} \quad (1)$$

$$f = P(\gamma |X) \quad (2)$$

$$T = \{ \gamma, P(\gamma |X)\} \quad (3)$$

Deep transfer learning is categorized into four categories [34], [35]:

1.1.2.1. Instances-based deep transfer learning

Instance-based deep TL refers to the case that a part of the instances from the source domain are re-used in the target domain and it was used initially based on AdaBoost model [36]. Although there are differences between the source domain and target domain, there are still a partial instance in the source domain that can be utilized by the target domain with appropriate weights [37]–[39]. Figure 1.3 shows a schematic map of instance-based transfer learning.

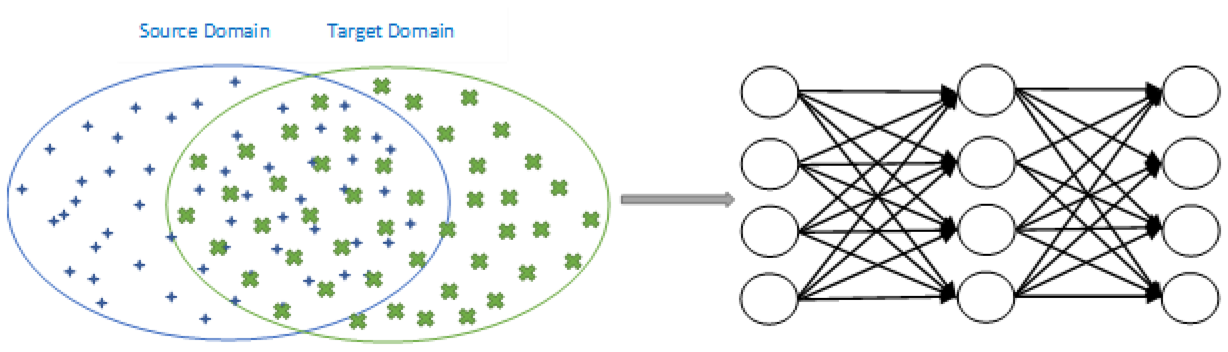


Figure 1.3. Schematic map of instance-based transfer learning

1.1.2.2. Mapping-based deep transfer learning

A map from instances in target domain and similar instances in source domain can create a new data space [40]. Creating this new data space called mapping-based transfer learning provides a similar and suitable domain to train a union neural network. Figure 1.4 demonstrates a schematic diagram of mapping-based deep transfer learning method.

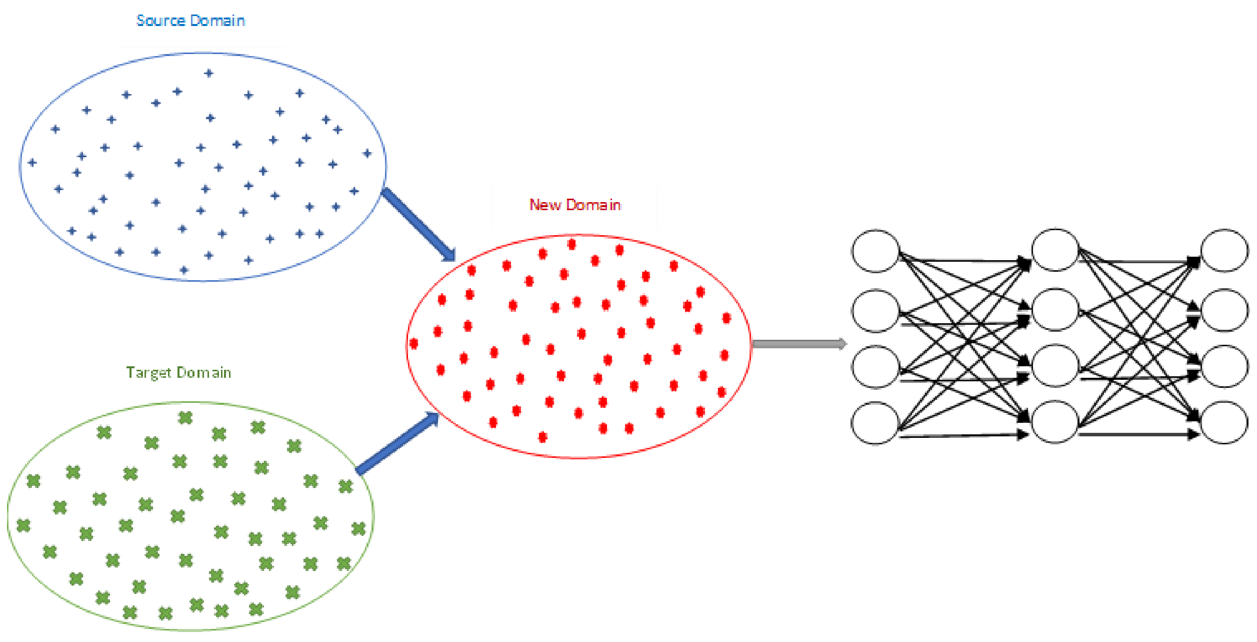


Figure 1.4. Schematic mapping-Based transfer learning

1.1.2.3. Network-based deep transfer learning

Network-based deep transfer learning refers to the approach that a part of another trained model is transferred and reused to a new ML model as a pre-trained model [41]–[43]. Specifically, all weights in the neural network layers in the source domain are adopted in the target domain before the training on the target domain begins [44]. Additional layers are necessary for the target domain for desired model performance. However, the inclusion of the pre-trained layers will dramatically reduce the training effort in terms of the number of labeled data and training time. **This is the approach used in this dissertation.** Figure 1.5 demonstrates network-based deep transfer learning model using another deep neural network as a pre-trained model and transfer weights from trained layers to train target domain.

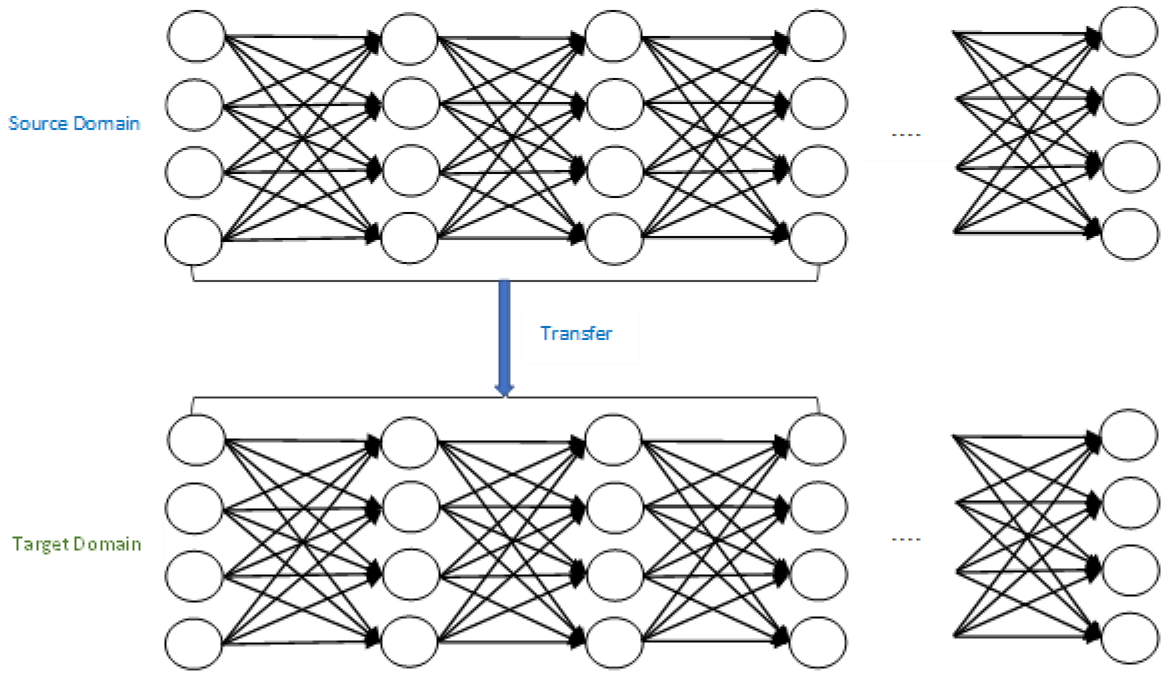


Figure 1.5. Schematic map of network-based transfer learning

1.1.2.4. Adversarial-based deep transfer learning

Adversarial-based transfer learning is inspired by generative adversarial networks (GANs) [45] to improve performance of generator, discriminator or both in a conventional GANs model. Different domain adaption methods such as domain adaption loss function or network-based transfer learning could be used for this purpose. There is no strong condition to determine how many images needed for training a DCNN model. As a rule of thumb, it is suggested that 1000 images per class should be sufficient. Note that this number could vary based on the problem. A DCNN model where there is approximately the same amount of data for each task may still benefit from transfer learning if there is a risk of overfitting, as it often occurs when the destination task is highly domain specific [30], [32]. In fact, training a large domain specific DCNN might be counterproductive, as it may overfit the domain. Overall, when used appropriately, transfer learning will provide a trifecta of benefits: a higher starting accuracy, faster convergence, and higher asymptotic accuracy (i.e. the accuracy level to which the training converges) [33], [46], [47]. It is not easy to locate TL among other ML methods since there is an overlap between each subcategory, but Figure 1.6 demonstrates the relation between TL and other ML methods.

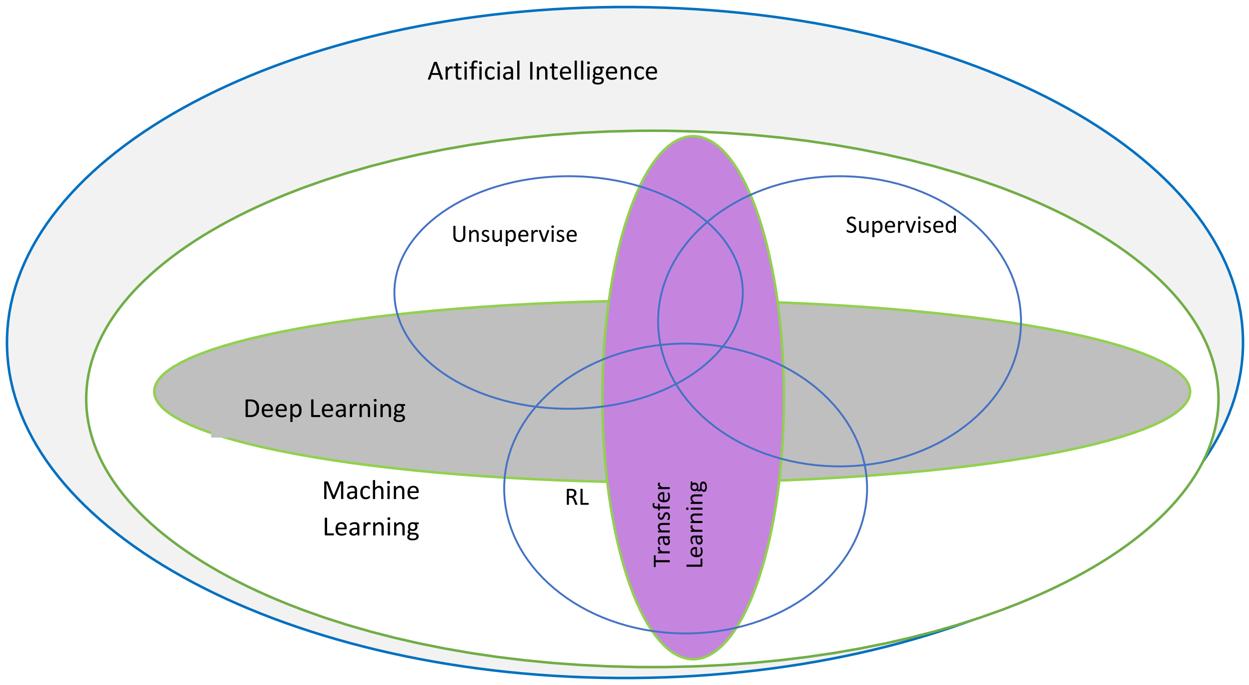


Figure 1.6. Relationship of TL (in purple) among other ML methods

This dissertation focuses on the applications of image segmentation using ML models applied to manufacturing process monitoring and agricultural field management. Table 1.1 summarizes the most relevant studies related to the applications of TL in manufacturing process monitoring. TL models might be affected by many parameters including the ML methods that the source model (pre-trained model) was trained on or the source or target dataset. Also, process monitoring could be done by different types of data such as text, images, or time series data. It is not easy to compare studies from different data types. This shows the wide gape and research potential of using TL methods in manufacturing and natural resource management.

Table 1.1. Most relevant transfer learning approaches for manufacturing process monitoring purposes

Study	Year	Case Study	ML Approach		Target Data Type	Datasets		Number Of Classes	Approximate Max Accuracy (%)
			Source	**Target**		**Source**	**Target Size**		
[48]	2018	Castings	ResNet	CNN	Image Segmentation & Object Detection	ImageNet	2500	2	95
[49]	2019	3D printing	transfer component analysis (TCA)	SVM	Time Series	vibration acceleration signals	2400 × 162	16	86-92
[50]	2020	Wind turbine	CNN	CNN	Time Series	Time waveforms of collected vibration data	12000 &24000	1&3	98
[51]	2021	Weld penetration	ResNet	CNN	Image Classification	ImageNet	28494	3	96
[52]	2021	Acoustic emission	VGG16	CNN	Image Classification	ImageNet	480000	12	92

According to Table 1.1, various ML approaches have been used in different studies. This research also used traditional ML in chapter two, DCNN in chapter three, and TDCNN in chapter four. Figure 1.7 demonstrates position of this study among other ML sub-categories in green color.

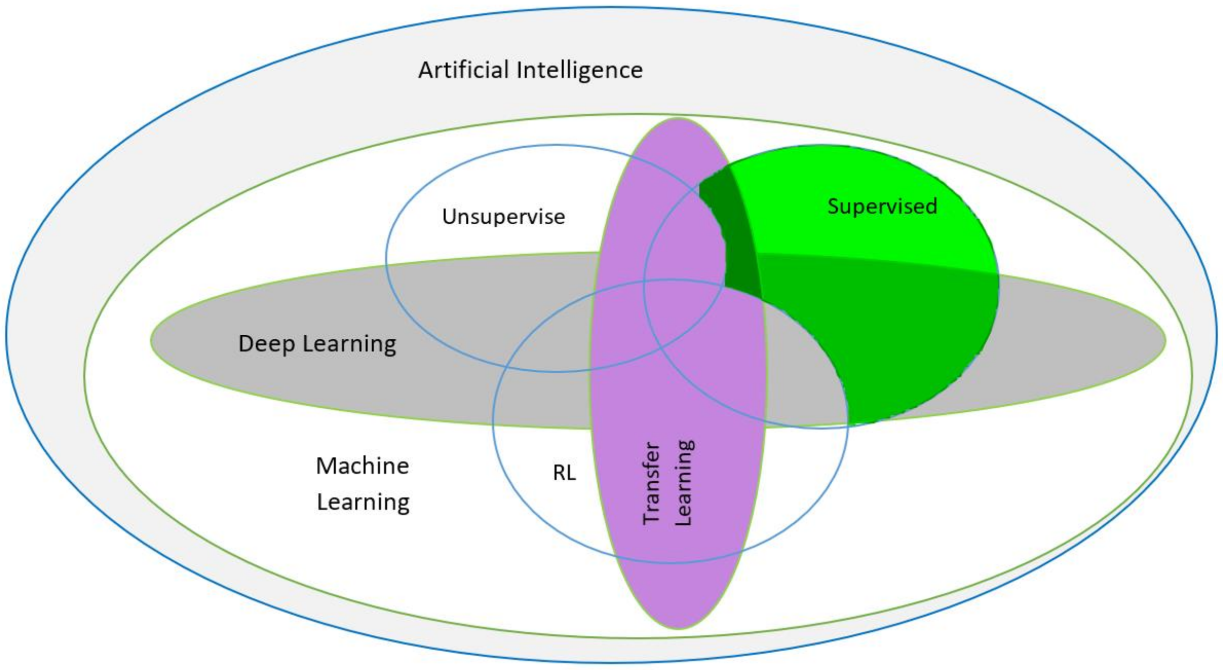


Figure 1.7. Relationship of TL (in purple) among other ML methods and position of this study (in green) under ML category

1.2. Challenges

Pre-processing (especially labeling images) might may be the most time-consuming and costly step in image processing. Traditional supervised ML methods for image classification are costly because of the need to label images manually. However, when objects have a very similar features, current unsupervised learning will fail to detect, segment, and locate them in the image. Supervised ML methods such as decision tree and other methods based on various trees such Random Forest (RF) or Gradient Boosting Classifier (GBC) usually provide good accuracy for prediction, but they have limited performance especially in the face of complicated problems. Complexity of a simple decision tree is in the order of $O(n^2p)$, for RF $O(n^2pn_{trees})$, and for Support Vector Machin (SVM) is $O(n^2p + n^3)$ where n is the number of training samples, n_{trees} is the number of trees for methods based on various trees, and p is the number of features [52]. Therefore, if the number of images and features increases, the running time for model training will increase exponentially.

Deep learning (DL) methods provide an alternative framework for feature extraction and classification that overcome some of the limitations of traditional machine learning approaches. Applying traditional machine learning (ML) methods, users need to design features manually, which is a serious burden for model building. In contrast, DL methods, such as deep convolutional neural network (CNN), use unsupervised, supervised, semi supervised, or hierarchical methods to achieve feature extraction automatically [53]. The total complexity of all convolutional layers is in the order of $O(\sum_{l=1}^{d} n_{l-1}\, s_l^2 n_l m_l^2)$ [54] where the index of a convolutional layer is *l*, number of convolutional layers is d, number of filters (also known as "width") in the lth layer is n_l. Also, n_{l-1} is also known as the number of input channels of the *l*-th layer. s_l is the spatial size (length) of the filter and m_l is the spatial size of the output feature map.

However, at least hundreds of labeled images are needed, and labeling is often required to have a well-trained CNN. Labeling this these many images is very time- consuming. Semi-supervised learning [55], where a small set of labeled images is used to classify a large dataset, usually provides better accuracy than unsupervised learning but is not as good as supervised learning methods. Another area in Machine Learning (ML) along with supervised and unsupervised learning is Reinforcement Learning (RL) where a software agent rewards different actions and maximizes total rewards based on a Markovian Decision Process (MDP) [56]. RL methods are promising for some applications such as video games [57]. However, an RL algorithm usually requires even more data than traditional ML and DL methods[58]–[62].

Despite the vast progress in developing novel ML/DL/RL models and their sharply growing applications, there are still several prominent challenges. These challenges include but are not limited to:

1- How can the burden of pre-processing and labeling be reduced for supervised ML models which demands a high level of costs and human resources?

2- Is it necessary to start a model-training process from scratch for a new ML problem?

3- How can process monitoring be implemented before a ML model is successfully trained?

4- How can ML model training time be reduced for a complex problem? The assumption is that solving more complex problems demands higher computational power and running time.

5- Is there a way to reduce complexity and improve accuracy at the same time for ML model building?

Transfer Learning (TL) is a potential solution to address all of the questions asked above. A new approach to answer these questions is called TL where uses the knowledge gained from a different and yet similar problem is used to solve another problem.

1.3. Dissertation outline

The remainder of this dissertation is organized as follows. Chapter 2 describes the application image processing and ML in statistical process monitoring [63]. This chapter involves an image-based quality monitoring framework capable of monitoring 3D printing parts layer by layer. The proposed framework is illustrated by a 243-layer basket part with 3-inch diameter and 1.5-inch tall. An overhead camera takes images after a printer finishes each layer. The traditional SPC phase I process requires at least 20 to 25 parts to establish control limits and each layer requires one set of control charts [64]. Since the material and resources using in 3D printing is usually expensive if a human operator needs to watch the printing process, the first challenge is how to monitor the printing process from the very beginning when there are not enough samples to establish these control charts. as well as the second challenge is how to automate process monitoring. Thus, the proposed framework first starts with a self-start control chart over based on layer-wise images in the beginning. To improve detection power and reduce false alarms, the proposed process monitoring method and switches to a cluster-charting approach after enough good parts are printed.

Chapter 3 adopts a DL model for a machine-vision application of soil coverage in agriculture fields. Unlike a manufacturing process that often takes place in a controlled and protected environment, agriculture applications, such as soil conservation and water management, often contain more variations [65]. Estimating crop residue and green canopy cover in agricultural fields is a key element in conservation agriculture and crop development. Current image analysis methods in agriculture are mostly limited to classifying a few types of crop residue images at the experimental level or just quantifying green canopy cover. This chapter aims to explore a deep CNN model for quantifying the percentage of stubble, live vegetation, and bare soil in downward-facing images of agricultural fields. The proposed tool based on this CNN model has features for easily and accurately quantifying green canopy cover, counting plants, and classifying stubble using a comprehensive dataset containing 3942 labeled images from real agricultural fields.

Finally, chapter 4 advances data-driven machine learning models in a manufacturing process [66]. Specifically, TL methods are explored to reduce the amount of data needed for training as well as to improve the accuracy of image analysis for a laser scribing process. To do so, image-based characterization of laser scribing quality using Aa deep transfer learning model is used for the identification of several quality characteristics such as debris, scribe width, and straightness of a scribe line. Images taken from the laser scribes on intrinsic Si wafers are examined. These images are labeled in a large and a small dataset, respectively. The large dataset includes 154 images while the small dataset only includes 21 images.

2. A layer-by-layer quality monitoring framework for 3D printing[1]

Abstract

Technology development in additive manufacturing is rapidly transitioning from mass production to mass customization. In this transition, automation in all stages of production including quality control is a key. In this study, a layer-wise framework is proposed to monitor the quality of 3D printing parts based on top-view images. The proposed statistical process monitoring method starts with self-start control charts that require only two successful initial prints. Answering the challenges of image processing due to lighting, a Machine Learning (ML) method is adopted to separate each layer from the printing bed. A sample image is compared to the standard image from a good part at each layer. The number of pixels in the difference images is fed into the proposed control charts to monitor the printing process at each layer. An Exponentially Weighted Moving Average (EWMA) chart based on the number of pixels is used for process monitoring at each layer. Once enough parts have been printed, homogeneous layers are clustered to reduce the number of control charts needed for process monitoring. Experimental results based on a 3-inch diameter basket part show that the proposed framework based on continuously monitoring of layer-by-layer images is able of detecting small changes in the printing process.

[1] Reprinted with permission from "A layer-by-layer quality monitoring framework for 3D printing" by Mohammad Najjartabar Bisheh, Shing I. Chang, and Shuting Lei, 2020. Computers & Industrial Engineering, 157, 107314.

2.1. Introduction

The ability of Additive Manufacturing (AM) often called 3D printing generating parts with complex geometry makes it a viable manufacturing option. AM is recognized as a disruptive technology in the digital innovation ecosystem [67]. The disruption is leading manufacturing paradigm evolution from mass production and lean manufacturing to mass customization [68], [69]. The global 3D printing industry revenue has been increasing from around one billion dollars in 2009 to over five billions in 2015 [67], [70], [71].

3D printing technology and mass customization has been used widely from automotive to food, healthcare and medical, fabric and fashion, and electric and electronic industries [72]. For example, Ford is the leader in car industries using 3D printing to produce prototype and engine parts [73], BMW uses 3D printing to produce hand-tools for automotive testing and assembly, and Audi uses it to produce spare parts [74]. Some companies such as Nike, Adidas, and New Balance provide in the traditional store and online services where customers can customize their own shoes by selecting from different varieties of material and designs and print them out [75], [76].

ISO/ASTM 52900 defined AM as "a process of joining materials to make parts from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing and formative manufacturing methodologies, ranks high on the transformative scale" [77]. In order tTo reach the full potential of AM, quality monitoring during a printing process is crucial to ensure production efficiency and customer satisfaction [78]. However, most 3D printers are not equipped with an automatic quality monitor feature. The lack of supervision may result in wasted prints especially if the printing problems occurred in early layers [79]. Machine vision provides a mean for several different applications such as test mining or an automatic process monitoring operation [80]–[82].

Automatic process monitoring is very important for large scale 3D printing operations in which hundreds of 3D printing machines are used in mass production of the same part. Even for small-batch 3D printing jobs, automatic process monitoring is also important in that there are not many printed parts for implementing traditional statistical process monitoring methods. A phase I control charting may not be established when the batch size is less than 20 parts. Support Action for Standardization in Additive Manufacturing (SASAM) drafted additive manufacturing roadmap and standards for quality and performance is one of the five high priority areas [83].

Due to the rich process information that can be captured, images and videos are increasingly deployed to monitor an AM process. Machine Vision System (MVS) can be used to inspect quality characteristics in many different industries such as liquid crystal display, ceramic tiles, textile, and food products [11]. Typically a MVS usually include a device to capture images (e.g. a vision sensor or cameras) and a computer to analyses and process the images captured by devices [84]. MVSs have been increasingly used in industrial process monitoring due to the efficiency increasing and cost reduction [13]. However, high-dimensionality, correlation structure and complex data characteristics present many challenges for existing process monitoring methods to fully utilize the information of color images [12]. By integrating MVS and statistical process control (SPC) in manufacturing process monitoring, not only product quality can be monitored, but also information gleaned from product images can be used for diagnostic analysis [13], [85].

The rest of this study is organized as follows. Section 2.2 provides some research background of imaged base quality monitoring with focus on additive manufacturing. Section 2.3 is devoted to two subsections of image preprocessing and proposed process monitoring framework. Then an experimental example is presented to highlight how the proposed methodology can be applied to a real-world problem in section 2.4. Finally, conclusion and future research can be found in section seven 2.5.

2.2. Literature review and contribution

A wide variety of quality characteristics such as product geometry, surface analysis, dimensional data, and defect patterns can be monitored using imaged based monitoring [11]. Image-based quality monitoring includes the following steps: image acquisition, image pre-processing, feature extraction, and process monitoring, and control charts [12]. Previous research in this area mostly focused on two approaches [13]. The first approach uses spatial control chart to detect the location of and size of a defect [14]–[18]while the second approach aims to detect defects through a well-defined statistical method [19], [20]. These two approaches may be implemented simultaneously [21].

The idea of using control charts to monitor image data was first proposed by Hosrt and Negin [86] for dimensional control of web production processes [86]. They showed that the use of control charts with image data improved productivity and profitability significantly. Koosha *et al.* applied imaged-based SPC for nonparametric profile monitoring [87]. They monitored the coefficient of extracted features with a generalized likelihood ratio (GLR) control chart. Their results under different fault tests showed that their model was able to detect shifts quickly.

Delli and Chang proposed a Support Vector Machine (SVM) method to distinguish good parts from bad parts based on segments on printed images at critical checkpoints [88]. Each image was first segmented into an 8 x 8 lattice forming 64 stamps. RGB statistics of these stamps were fed into the proposed SVM for a go/no-go determination. A major drawback of this method was that a large number of training images including both good and bad parts were required. Often time the bad parts were hard to collect and might take a long time before this proposed process monitoring method could be implemented.

Imani *et al.* used a layer-by-layer image analysis to detect the onset process conditions that led to a lack of fusion-related porosity from in-process sensor data in laser powder bed fusion additive manufacturing [89]. Their goal was not only to quantify size, number, and location of pores but to identify process conditions that were liable to cause porosity through analysis of in-process, layer-by-layer images. They successfully examined and showed that several machine learning techniques could be used to detect pores in laser additive manufacturing. Imani *et al.* implemented a deep learning neural network in layer-wise imaging profile for additive manufacturing quality control [90].

In reality, there might be several layers of printing and quality features in 3D printing parts which could generate a large amount of data which was hard to track and monitor. Zou *et al.* proposed an Exponentially Weighted Moving Average (EWMA) and region growing based control chart to monitor images of production lines where the quality characteristic had either a specific pattern or uniformity character [91]. Table 2.1 summarized position of our proposed method among other researchers' work. As we can see in Table 2.1, most existing methods in statistical monitoring wait until the finished print for process control. The proposed method works in a layer-by-layer fashion so that remedial actions can be taken while a part is still printing. Finally, the methods which are capable of layer-by-layer process monitoring (such as [89], [90]) requires a very large amount of image samples to implement. On the other hand, the proposed method can start with a minimal number of images.

Table 2.1. Related Imaged-based Quality Monitoring Methods

	PAPER	YEAR	CASE OR INDUSTRY	FEATURE EXTRACTION (ML) AND/OR MONITORING METHOD	LAYER-WISE MONITORING	GROUPING LAYERS	SELF-START PROCESSING
1	Lu & Tsai	2005	Thin film transistor liquid crystal displays (TFT-LCDs)	Singular Value Decomposition (SVD)	No	No	No
2	Lin & Chiu	2006	Liquid Crystal Displays (LCD)	Hoteling T^2 Statistics, Ant Colony Algorithm, Back Propagation Network	No	No	No
3	Megahed et al.	2012	nonwoven textile	GLR	No	No	No
4	He et al.	2015	Lego car manufacturing lab	Multivariate GLR	No	No	No
5	Yan et al.	2015	steel tube	Unfolded Principal Component Analysis (UPCA) and Low-rank tensor decomposition (LRTD), Hoteling T^2 control chart	No	No	No
6	Koosha et al.	2017	nonwoven textile	GLR	No	No	No
7	Delli & Chang	2018	3D printing	SVM	No	No	No
8	Imani et al.	2018	Laser Powder Bed Fusion	Multifractal and Spectral Graph Theory, Neural Networks (NN)	Yes	No	No
9	Imani et al.	2019	Laser Powder Bed Fusion	Convolutional Neural Network (CNN)	Yes	No	No
10	Zuo et al.	2019	nonwoven textile	Region Growing Algorithm, EWMA	No	No	No
11	**Proposed Method**	**2021**	**3D printing**	**NN, SVM, and Gradient Boosting Classifier (GBC), EWMA**	**Yes**	**Yes**	**Yes**

In this study, an image-based quality monitoring framework is applied to 3D printed images layer by layer. The proposed framework is illustrated by a 243-layer basket part with 3-inch diameter and 1.5-inch tall. An overhead camera takes an image after a printer finishes each layer. The traditional SPC phase I process requires at least 20 to 25 parts to establish control limits and each layer requires one set of control charts [64]. Since the material and resources using in 3D printing is usually expensive, the challenge is how to monitor the process from the very beginning when there are not enough samples to establish these control charts as well as how to automate process monitoring.

Thus, the proposed framework first will start with a self-start control chart over layer-wise images in the beginning and switch to a cluster-charting approach after enough good parts are printed. The cluster-charting approach includes an Auto Regressive Integrated Moving Average (ARIMA) filter to alleviate the autocorrelation of statistics from adjacent layers and then use one EWMA control chart for each homogenous layer family.

2.3. Problem description and proposed method

In 3D printing industry, quality of 3D printed parts is checked when a part is printed. This practice means that if there is a mistake even in the first layer of printing, not only it can cause a significant waste in material and time but also the part should be printed again. Figure 2.1 (a) and (b) show two defective samples that can happen in a 3D printing process toward the end of a printing process. Figure 2.1 (c) shows another defect when operator discovered the problem at an early stage and stopped the printing process. This study attempts an automatic inspection framework to monitor quality of parts after each layer of printing by taking pictures and compare them with a standard print image.

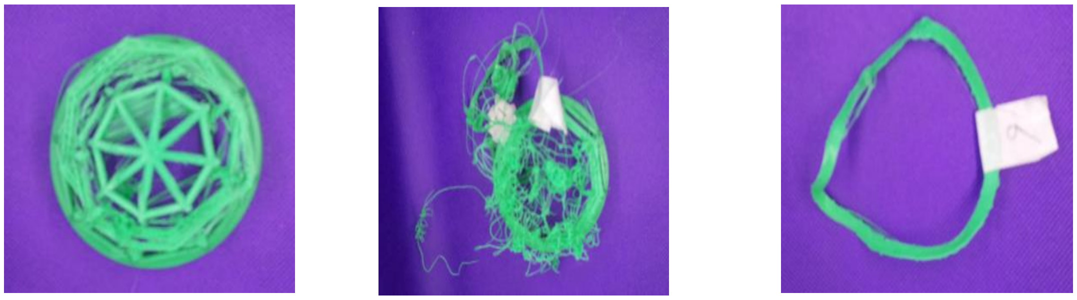


Figure 2.1.(a) Defect at the end of process **(b)** Toward final layers of printing **(c)** Defect during early stage of printing

The proposed method starts with data preparation and image processing and then goes to process monitoring. Figure 2.2 shows the proposed imaged-based quality monitoring framework. In image processing, images will be cleaned, classified, and prepared to monitor the process. By having cleaned and classified images, process monitoring will start. Monitoring of the process includes two methods. First method (self-start control charting) is designed in the beginning of the printing when there are not enough parts to design a regular control chart.

In this approach in order to find inverse normal distribution (U_{nj}) at least two printed sample is needed. After printing two standard parts, the proposed framework will start by feeding image of first layer to monitor quality of the process. The basic assumption is that the quality of the two printed parts is satisfactory. This process will be implemented for each layer of printing separately. Each step is explained in more details. Note that in 3D printing industry there is a high chance that there is a demand just for less than a hundred parts. Thus, the self-starting might be the most helpful method in 3D printing quality monitoring.

For the scenario where there are several hundred parts, we can use the second proposed method after printing the first 20-25 parts. In this case, a regular EWMA control chart can be used. Another major contribution of this study is the ability to monitor the process layer-by-layer so that the process monitoring is accomplished before a part is printed. However, the main challenge is that two images from two consecutive layers might be 99 percent similar. This similarity implies the possibility of autocorrelation in the time series feeding into a control chart.

To remove autocorrelation, an ARIMA filter is used and then data is standardized. Another major challenge is homogeneity. Information gleaned from in the beginning layers may not be the same as those in the final layers. Thus, to overcome this problem and reduce false alarms, Sullivan's change point detection [92] is used here for creations of homogeneous clusters.

Sullivan's method is designed to capture mean shift. In our study, we used Sullivan's method to group consecutive homogeneous layers together. One control chart can be implemented for each homogeneous cluster.

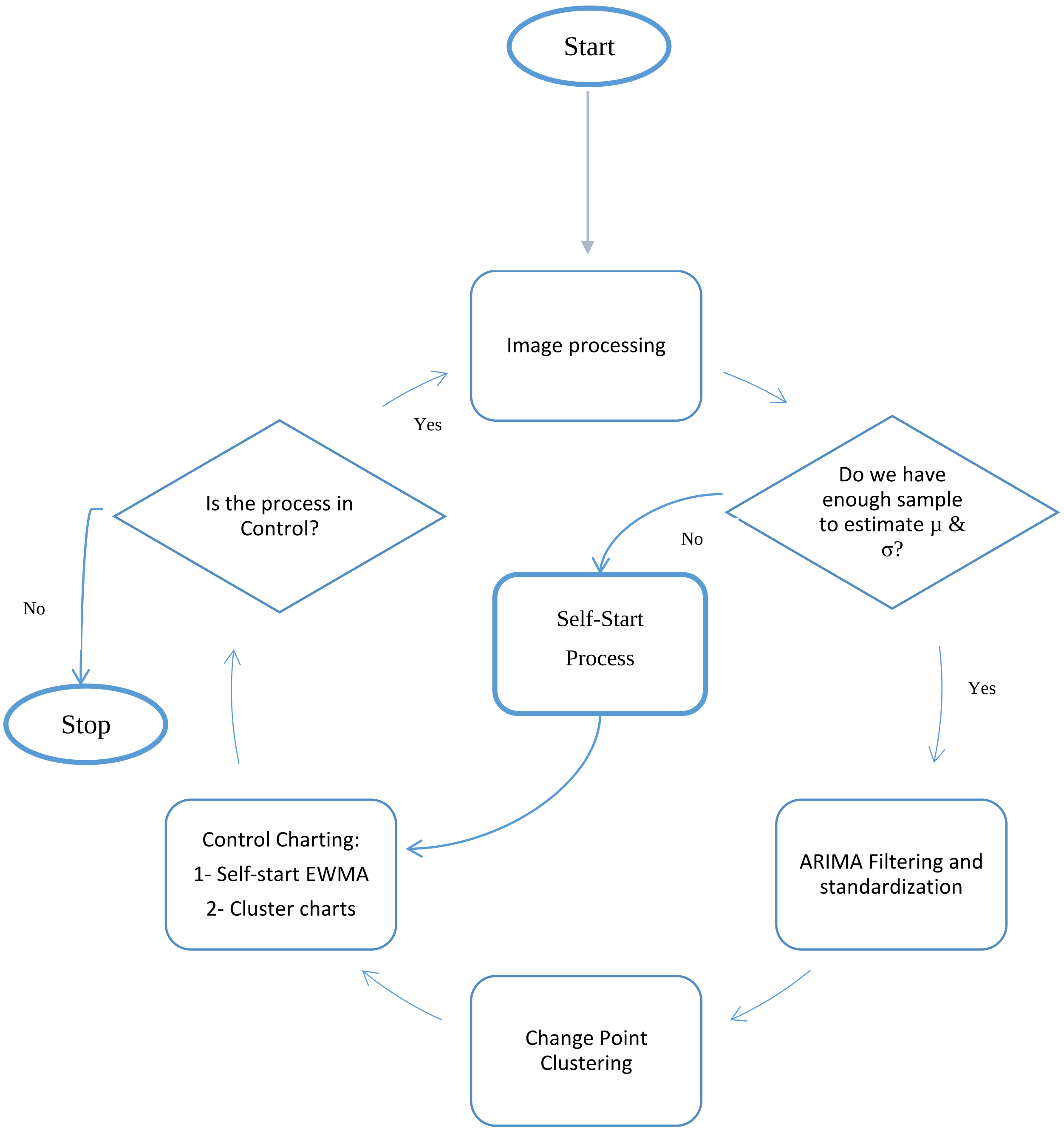


Figure 2.2. Layer-by-Layer Image-Based Quality Monitoring Framework for in-process 3D Printing Parts

2.3.1. Image processing

Image processing is the first core stage in the proposed quality monitoring framework. To do this, RGB values for each pixel in the image is compared to the same pixel location in the corresponding standard image. A gray-scale image can be represented as a function f (x, y) where x and y can take non-negative integer values. For an 8-bit unsigned integer, this value is between 0 (black) and 255 (white) and the same color for a 16-bit unsigned integer is between 0 and 65535. However, printed layers are a small proportion of the printing bed as shown in Figure 2.3 and a small change in lightening may cause a large difference in RGB values.

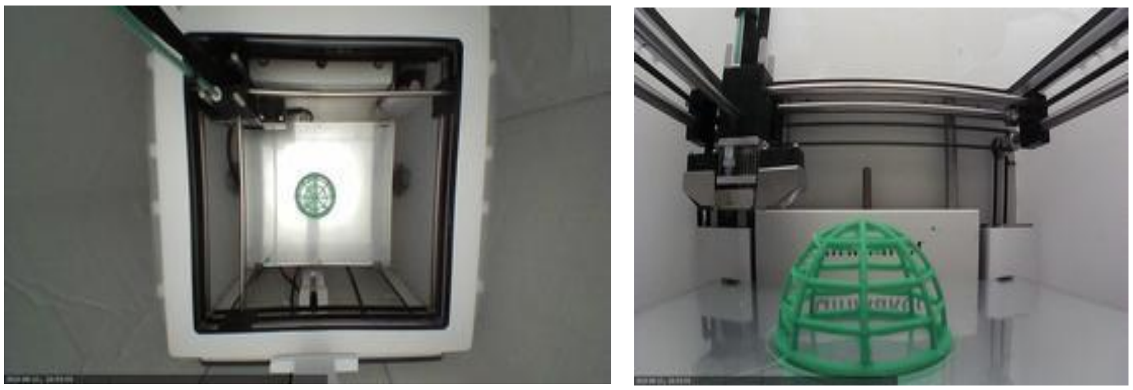
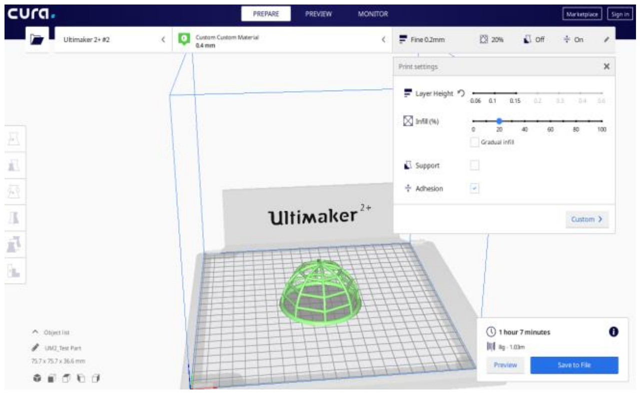


Figure 2.3. Experimental setup

To identify a printed layer from the printing bed, the proposed method first classifies each pixel into two classes: a part of the print or not a part of the print. This way, the part image is isolated from the machine bed and its environment so just the parts would be compared and not the environment. Then the filtered image containing only the part can be compared to a standard image. Two quality monitoring techniques in the proposed framework are then applied on the difference of sample and standard images to determine whether the process is in control or not. In this section, data preparation and image processing of our study is explained step by step.

2.3.1.1. Experimental Setup and Image Capturing

The 3D printer used in this study was Ultimaker 3. Ultimaker Cura was used to design the part and to handle the G code, remote access to the camera feed, and mage webcam snapshots. Octoprint software served as the operation platform was used to print a part and collect part images layer by layer. Figure 2.4 shows that the part design on Ultimaker Cura. Octoprint can be implemented in either on a PC or Raspberry Pi. To ensure consistent lighting, an enclosure was built to cover the Ultimaker printer. A strip lighting and a photography umbrella are mounted on the top of the enclosure. After printing of each layer, the G code instructed the printing head to its default location at the upper left-hand corner shown in the top view picture in Figure 2.4. Then the overhead webcam takes a picture. The material used for this research was PLA. The same 3D printer configuration was used throughout this study. Finally, Python 3.7 has been used for image processing and extraction.



Ultimaker 2+ #2
Custom Custom Material 0.4 mm
Fine 0.2mm 20% Off On
Layer Height: 0.06 0.1 0.15 0.2 0.3 0.4 0.6
Infill (%): 0 20 40 60 80 100
Gradual infill ☐
Support ☐
Adhesion ✓
UM2_Test Part
75.7 x 75.7 x 36.6 mm
1 hour 7 minutes
8g · 1.03m

Figure 2.4. The part designed using Ultimaker Cura

Images were taken by two cameras – from the top (Figure 2.3 (a)) and from the front (Figure 2.3 (b)) of the Ultimaker printer. In this study, we focus on the data only from top view camera. Side view images can be processed and analyzed using the same methods. In addition, we are working on using several cameras from the corners so that the printer head does not have to travel to the default location after each print. Figure 2.4 presents a possible angle for this implementation.

2.3.1.2. Image Pre-processing and Filtering

To make sure suitable images are generated with pixels containing part only, it is essential to clean the data. First, the portion containing the part in Figure 2.3 (a) was cropped to contain only the printed layer and its surrounding bed area. Then all the pixels in the cropped image need to be classified into two classes: part only and printer bed. Finally, each pixel containing the RGB values is transformed from colored images into binary images with “part” and “not part” label. In other words, the proposed approach first converts the RGB values in each pixel into a gray scale value. Then a gray scale cut-off threshold value can be used for the binary classification task[93]. However, lighting is a significant issue as shown in Figure 2.5 where the reflection of the part on the printer bed causes some of the printer-bed pixels to be classified as a part of the part. A single threshold value cannot be implemented for the classification task.

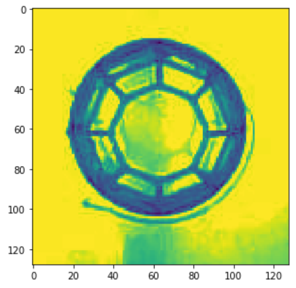
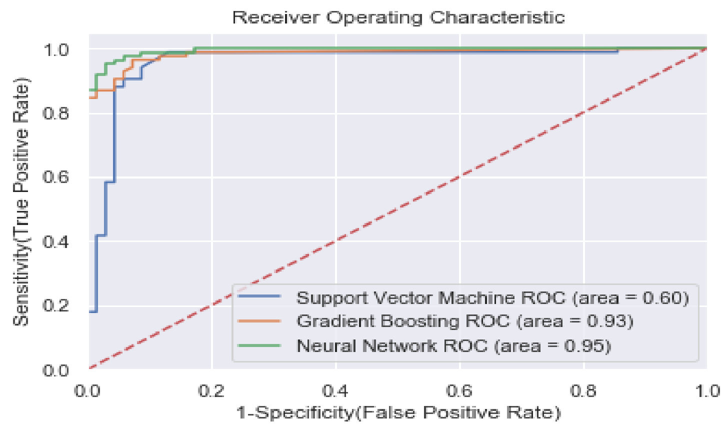


Figure 2.5. RGB to gray scale using skimage package on python

To overcome this issue, three different machine learning algorithms of Neural Network (NN), Gradient Boosting Classifier (GBC) and Support Vector Machine (SVM) from sklearn package in python were used to achieve the classification task. These methods were chosen because of their performance in binary classifications. NN and SVM are two well-known machine learning algorithms in binary classification and GBC is a newer and improved method based on random forest. Totally 6000 pixels include 3000 part pixels and 3000 non-part pixels from five different images has been picked and labeled for training purpose. The labeled data then split to training, validation, and test set with 60 percent for test, 20 percent for validation and 20 percent for testing.

Among three machine learning methods studied, SVM which had the worst performance while NN and GBC performed more than 90 percent accuracy. NN had slightly better performance than GBC in terms of True Positive (TP) and False Positive (FP) Rate, but GBC had lower FPR which is more critical to eliminate the part reflection on the printer bed mistaken as the part in Figure 2.5. So in this research, GBC is chosen for further analysis. Figure 2.6(a) shows the AUC curves of the different ML methods studied. Comparing various ML methods, we prefer AUC curves trending toward upper left-hand corner. The preliminary data shows that NN and GBC are capable of detecting part with 95 and 93 percent accuracy while SVM can only achieve 60 percent accuracy for pixel classification.



		Predicted Values	
		Positive	Negative
True Values	Positive	TP = 853	FP = 33
	Negative	FN = 37	TN = 888

Figure 2.6. (a) ML Performance Sensitivity (b) Confusion matrix for GBC model

The GBC method was used for pixel classification and Figure 2.6(b) shows True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) of the confusion matrix over test set for GBC. GBC was chosen because of the highest accuracy over test set as well as lowest false negative value among other methods. This process has to be done for all the layers in every single part. The sample part in this study has 243 layers. Figure 2.7 depicts the outcome for layer 200 of the first part. Figure 2.7(b) is the color image while Figure 2.7(a) is binary image where the light color (i.e. yellow) represents “part” and the dark color (i.e. purple) is the printer bed. Due to the classification errors, the filtered image contains noise and loses resolution. As shown in the outer rim in the original image almost disappears in the filtered image. This sacrifice is necessary in that each pixel contains three values in Red, Green, and Blue, which cannot be directly used in the next stage of the proposed method, which is the image difference operation.

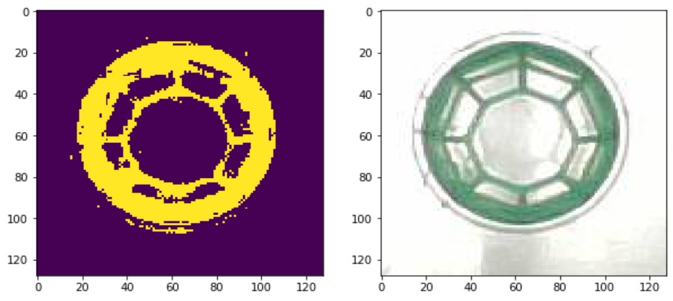


Figure 2.7. (a) GBC binary prediction on the left and (b) the actual part (Layer 200) on the right

2.3.1.3. Image Extraction

The outcome in the previous section is a gray-scale filtered image free of noise from lighting and background as shown in Figure 2.7(a). The proposed imaged-based process monitoring framework involves a layer-by-layer monitoring scheme. Specifically, any future sample layer would be compared to a good layer, which can be established when the first two good print is obtained. Each grayscale image from a given layer is represented in a two-dimensional matrix and resolution of each image is 128 x 128 in this case. To compare images, the 2D grayscale matrix of the sample layer should be subtracted from that of the first good layer. The pixels not containing any part (i.e. the printer bed) are already assigned the value zero by the proposed GBC algorithm in the last section while pixels containing the part are assigned the value 255.

Consider the case of n parts each with m layers. Let M_{ij} be the image matrix, i=1,2,…, n and j=1,2, …, m and its resolution is 128x128. The element in the M matrix can be presented as M (h, v) where horizontal index h=1,2,…, 128 and vertical index of v=1,2, …, 128. For example, M_{1j} is the image of the first part at jth layer. Define D_{ij} as the total number of different pixels for the jth layer between ith part and the first standard or good part.

Thus:

$$D_{ij} = \sum_{h=1}^{128} \sum_{v=1}^{128} \frac{|Mij(h,v) - M1j(h,v)|}{255} \quad \text{where i=2, 3, …, n and j=1, 2, …, m, h=v= 1, 2, …, 128.} \quad (1)$$

Remember that M_{ij} is a matrix containing elements with zeros representing “no part” and 255 representing “part.” Thus, the expression “$Mij(h,v) - M1j(h,v)$” is a value of either 0 or a multiple of 255. When an element in D_{ij} is 255, it means that there is a difference in the sample image pixel and that of the first part at layer j. To find total different pixels we need to divide it by 255 and tally all elements in the matrix M_{ij}.

2.3.2. Process monitoring

D_{ij} generated in equation (1) can be used for the process monitoring purposes. The goal of process monitoring is to ensure print quality at each layer as opposed to at the end of an entire part. Two complementary approaches are explored to monitor the production process depending on the amount of information available for Phase I of control charting. Traditional Phase I control charting guideline demands at least 20 to 25 observations to establish a control chart. However, Hawkins suggests the use of a self-start control charting procedure when this number can be reached [94]. The first method applies one self-start control chart to each layer. The proposed approach can start from the third part after two parts are successfully printed and the layer-by-layer images are stored. Equation (1) is implemented to generate the statistic for control charting. Since the part in this study contains 243 layers, we use 243 control charts each for a layer. The second approach can be implemented when enough homogeneous sample statistics are available. The core idea is to use as fewer control charts as much as possible by grouping adjacent layers. Instead of using one control chart for each layer, statistics from "similar" layers can be plotted on the same control chart. This requirement can be met after multiple parts are successfully printed and similar adjacent layers can be clustered according to the distribution of the standardized D_{ij}. The one control chart is applied to each homogeneous cluster.

2.3.2.1. Method I: Self-start Charting

A self-start control chart can be used for process monitoring on the number of different pixels in the matrix D_{ij}, j=1,2,…, m. Hawkins proposed self-start Cumulative Sum Control Chart (CUSUM) [94]. In their proposed method, any other control charts can be applied when standardized values for observation is calculated. CUSUM is an approach to catch small shifts [95]. However, in image-based quality monitoring we are mostly interested to catch medium to large size shifts. Thus, EWMA would be a better approach than CUSUM.

In this research we propose an EWMA self-start control chart to start monitoring of each layer from the third part. One control chart is applied for each layer of printing in this method and each point in the chart represents the quality at end of each layer. If the difference statistic D_{ij} plots within the control limits, it means that the process is in control. We could have chosen Individual

X control chart in this case since the processing monitoring task is mainly for various mis-printing situations such as those in Figure 2.1 (a)-(c). However, an IX chart requires the underlying statistics are normally distributed which is not true in this case.

Equations for this method are summarized as follow where $\overline{D}_{nj}$ is the average of the first n observations of D_{ij} and ω_{nj} the sum of squared deviations from $\overline{D}_{nj}$ for each layer j:

$$\omega_{nj} = \sum_{i=1}^{n}(D_{ij} - \overline{D}_{nj})^2 \tag{2}$$

$$\overline{D}_{nj} = \overline{D}_{(n-1)j} + \frac{D_{ij}-\overline{D}_{(n-1)j}}{n} \tag{3}$$

Thus, we can have:

$$\omega_{nj} = \omega_{(n-1)j} + \frac{(n-1)(D_{ij}-\overline{D}_{(n-1)j})^2}{n} \tag{4}$$

The sample variance of first n observation (s^2_{nj}), standardized observation (T_{nj}), cumulative t distribution of standardized observation ($F_{(n-2)j}(a_{nj}T_{nj})$), and the transformation of inverse normal distribution (U_{nj}) formula for $n \geq 3$ are given as follow:

$$s^2_{nj} = \frac{\omega_{nj}}{n-1} \tag{5}$$

$$T_{nj} = \frac{D_{ij}-\overline{D}_{(n-1)j}}{s_{(n-1)j}} \tag{6}$$

$$a_{nj} = \sqrt{\frac{n-1}{n}} \tag{7}$$

$$P(T_{nj} \leq t_j) = F_{(n-2)j}(a_{nj}T_{nj}) = F_{(n-2)j}\left(t_j\sqrt{\frac{n-1}{n}}\right) \tag{8}$$

$$U_{nj} = \Phi^{-1}[F_{(n-2)j}(a_{nj}T_{nj})] \tag{9}$$

Note that we assumed T_{nj} is normally distributed. A large value of D_{ij} means that huge difference between the printing part and first standard sample at layer j and a small number means the process is in control. So one can use values of U_{nj} in a CUSUM chart because in self-start we want to be careful about small and medium size changes. However, because of the lightening and environmental changes issues, and philosophy of CUSUM in catching very small changes, CUSUM for images-based quality monitoring might cause many false alarms. Thus we recommend the use of EWMA control charts for each layer j which is a better approach to catch medium to large size shifts by adjusting the EWMA parameter l toward 1. Table 2.2 shows a self-start procedure calculation for layer 122 of our printed parts and Figure 2.8 demonstrate self-start EWMA control chart for layer 122 of first 16 parts.

Table 2.2. Self-Start control chart calculation for layer 122 of the first 16 parts

Part number(n)	D_{n122}	$\bar{D}_{n122}$	ω_{n122}	S_{n122}	T_{n122}	$a_{n122}T_{n122}$	$F_{(n-2)122}(a_{n122}T_{n122})$	U_{n122}
1	347	347	0	-	-	-	-	-
2	272	309.5	2812.5	53.03301	-	-	-	-
3	266	295	4074	45.13314	-0.82024	-0.66973	0.312159661	-0.48974
4	246	282.75	5874.75	44.25212	-1.08568	-0.94022	0.223177095	-0.76151
5	242	274.6	7203.2	42.43583	-0.92086	-0.82364	0.235258149	-0.72164
6	311	280.6667	8307.333	40.76109	0.857766	0.783029	0.761313095	0.710533
7	228	273.1429	10684.86	42.19964	-1.29208	-1.19624	0.142615765	-1.06864
8	319	278.875	12524.88	42.29974	1.086671	1.016488	0.82569006	0.93727
9	275	278.4444	12538.22	39.58886	-0.09161	-0.08637	0.466795854	-0.08333
10	404	291	26726	54.49363	3.171487	3.008737	0.991577035	2.390053
11	446	305.0909	48566.91	69.68996	2.844369	2.712	0.988040471	2.258427
12	418	314.5	60253	74.01044	1.620163	1.551188	0.924048633	1.432843
13	398	326.6154	82735.54	83.0339	2.128078	2.044591	0.967208136	1.841258
14	403	303.2857	179319.1	117.4469	-3.93352	-3.79043	0.001287113	-3.01448
15	400	283.0667	263557.7	137.2062	-2.58232	-2.49476	0.013426394	-2.21365
16	472	290.25	274343.3	135.2389	0.837668	0.811069	0.784550315	0.787654

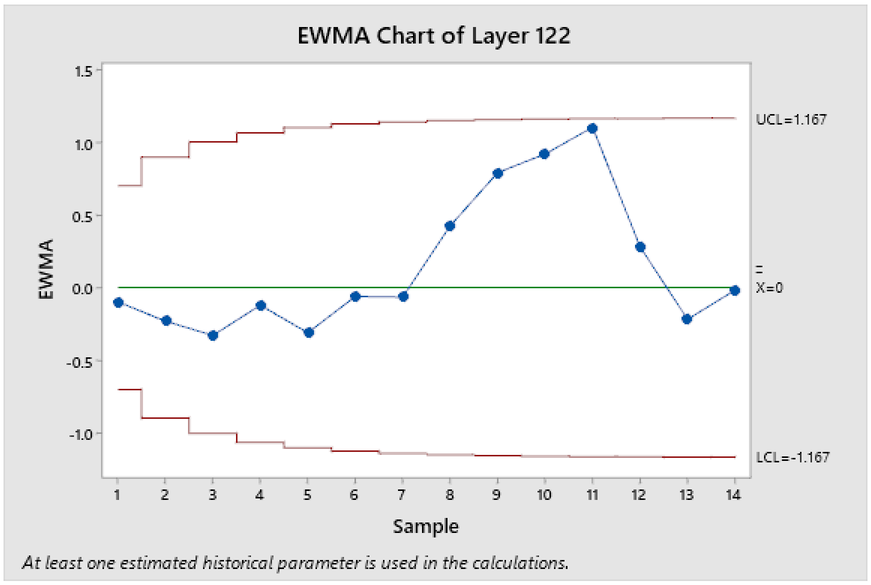


Figure 2.8. Self-start EWMA control chart for layer 122 of first 16 parts

2.3.2.2. Method II: Cluster Charting: Standardizing, Autocorrelation Filtering, Change Point Clustering

The self-start control chart is able to run real time process monitoring and detect defect. However, in self-start method for each layer one control chart needs to be monitored. For example, in our 3-inches basket case, there are 243 layers which means 243 control charts should be monitored. It would be desirable to have one or a few control charts for each part. This can be done when there are enough samples (20-25) to build a regular SPC control chart.

The cluster-charting method aims to reduce type II error by grouping adjacent similar layers. In 3D printing, comparing printed parts in the very beginning layer with the last layer based on just images might increase chance of error type II. This is because at the very beginning most of an image is bed of printer and we are analyzing pixels in an image that contain part and not bed of a printer. Thus, to overcome this problem and reduce false alarm probability, Sullivan's change point detection [92] is used here. Sullivan's method is designed to capture mean shift [96]. In our study, we used Sullivan's method to find mean shift in non-defect part. This will help us to know at what layer we have significant change in number of pixels include part and then monitor similar layers with same control chart.

Chang et. al. proposed a real-time detection of condensation-water-temperature wave profile monitoring. The core idea is to monitor product quality during the curing process rather than at the end of the process [97]. The same idea can be used in this application as well since 3D printing is accomplished layer by layer. Each layer represents a critical stage in the production process of interest. The print quality monitoring should take place at the end of each layer rather than at the end of the entire print. This method contains two major steps of standardization and ARIMA filtering and then change point clustering. Unlike the self-start control charting in section 5.1, we use much fewer number of charts for process monitoring.

2.3.2.2.1. Standardization and ARIMA filtering

The statistics in self-start control charts are independent because each point in a control chart comes from a different part. However, in the effort to plot statistics from different layers of the same part on the same control chart, the independent assumption may be violated. To test the i.i.d. assumption, we need to first standardize statistics D_{ij}.

$$e_{ij} = \frac{D_{ij} - \mu_j}{\sigma_j} \tag{10}$$

Note that in the self-start charting, control chart points are from different parts, i.e. i=1,2, … The plot statistics are independent since they from different parts. In the cluster charting method, we want to use the same chart family to plot e_{ij}, j=1,2, … The e_{ij} statistics, j=1,2,…,m, on the other hand, might be auto-correlated. Autocorrelation function plot (ACF) and Partial Autocorrelation function plot (PACF) can be used to test this autocorrelation. In order to remove the autocorrelation from e_{ij} we needed to apply a filter as in Figure 2.2. ARIMA can be used to remove autocorrelation from e_{ij}. If ACF dies out gradually and PACF cuts off sharply after a few lags, then an AR filter is recommended and if PACF dies out gradually and ACF cuts off sharply then MA filter is recommended. After the filtering operation, the uncorrelated is named e'_{ij}.

2.3.2.2.2. Change Point Clustering

From the previous section, we have removed autocorrelation from the statistics and generated statistics e'_{ij} which come from different layers of the different sample part. The next step is to cluster homogeneous streams of e'_{ij} for control chart families. Sullivan's change-point detection approach [92]is proposed for this task. This clustering algorithm for each part i find distance (d_{ik}) with m -1 boundaries (k_j) which separating the layers into clusters:

$$\bar{x}_{ik} = \frac{\sum_{j=1}^{k} \acute{e}_{ij}}{k} \tag{11}$$

$$d_{ik} = \frac{|\bar{x}_{ik} - \bar{x}_{i(k+1)}|}{s_i \sqrt{\frac{m_{ik} + m_{i(k+1)}}{m_{ik} m_{i(k+1)}}}} \quad i = 1,2, \dots n, k = 1,2, \dots , K \tag{12}$$

Where m_{ik} and number of layers in the adjacent clusters, s_i is an estimation of standard deviation of all clusters. Since the ranking of d_{ik} does not depend on the standard deviation so we can set it to one without loss of generality and $\bar{x}_{ik}$is the layers mean. Different parts might have different change points. In a typical 3D printing part, the difference from layer to layer may be very small. The implementation in the proposed self-start approach may be relaxed to allow adjacent layers to be combined. However, small incremental changes may accumulate to a large change. Therefore, it is necessary to cluster similar layers in term of the statistic e'_{ij} together. This procedure can be simply done by finding high density of change point around a specific layer. For example, if most of the parts exhibit their first change point between layer 57 to 62 then probably layer 60 can be a general change point for all the parts. Note that the number of change points might be different from part to part. Therefore, we pick the maximum number of change points among all parts to this procedure. This practice might require more control charts, but the number of charts is far less than the number of layers.

2.3.3. Control charting

By having values of U_{nj} in the first method (self-start charting) and e'_{ij} in the second method, we propose to use method I to start process monitoring after two acceptable prints are accomplished. After more successful parts have been printed, we will switch to method II. Since each printing part has different characteristics, the timing, and criteria for switching depends on how fast the estimates of μ_i and σ_i in equation (10) is stabilized.

In our problem we are mainly interested in catching medium to large shifts. EWMA control charts can be adjusted to meet these needs. Thus, in this case of self-start charting, EWMA control charts is used to plot U_{nj}. Each layer will be monitored with a control chart so 243 control charts will monitor the process. Self-start statistics are updated after every new printed layer. Same as the self-start charting, EWMA control charts can be used for each cluster. Note that estimation for standard deviation can be find by using $\frac{\overline{MR}}{d_2}$ where the $\overline{MR}$ is the average of moving range from part to part for a layer and d_2 is a function of sample size. Since adjacent layers are used, e_{ij} of adjacent layer are used to estimate the standard deviation, d_2=1.128.

2.4. Numerical results and discussion

The proposed framework adopts various methods such as machine learning techniques, ARIMA filtering, self-start control charting, and change-points-detection for clustering for image-based monitoring of 3D printing parts. It aims to automatically detect bad prints in every layer using the proposed self-start control chart which only requires the first two successful prints. In this study, we printed 15 non-defect part plus four defect parts to demonstrate the proposed methods.

2.4.1. Self-start EWMA control charts

Self-start EWMA control charts have been designed to monitor the production process after printing two parts. In addition to the 15 non-defect parts that we already used to design the traditional EWMA control chart and to see performance of the method, a 16th defect part has been added to the data set. Figure 2.9(a) and (b) shows layer 122 and 123 of part 16 respectively. With a naked eye it looks there is no difference between two images. If an operator checked the process after each layer, he might not be able to understand if one extra layer is printed or not. However, layer 123 is the start point of making a defect part. It is might not be clear from these images but after layer 122, the part had a small shift on the bed and that small displacement cause the huge mess which can be seen in Figure 2.9(c) as the last layer of this part.

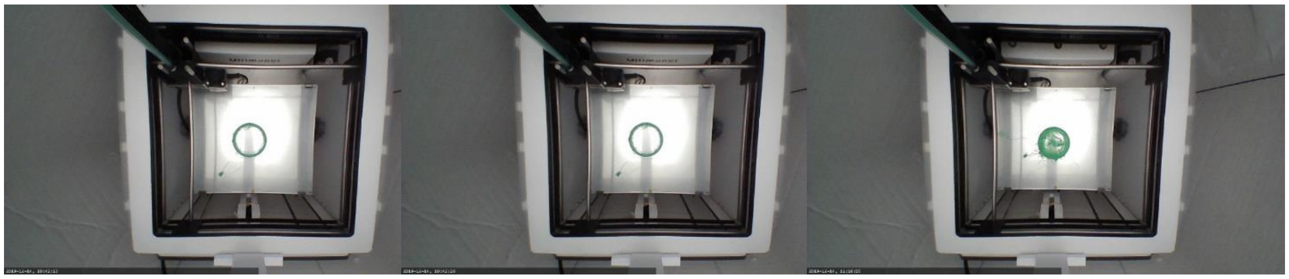


Figure 2.9. (a) Layer 122 of the 16th part (b) Layer 123 of the 16th part (c)Layer 243 of the 16th part

We already have seen self-start procedure for layer 122 of first 16 parts. To see how the proposed method, catch this problem, we only need to run self-start EWMA control chart for layers 123. Table 2.3 shows self-start procedure calculation for layer 123.

Table 2.3. Self-Start control chart calculation for layer 122 of the first 16 parts

Part number(n)	D_{n123}	$\bar{D}_{n123}$	ω_{n123}	S_{n123}	T_{n123}	$a_{n123}T_{n123}$	$F_{(n-2)123}(a_{n123}T_{n123})$	U_{n123}
1	322	322	0	-	-	-	-	-
2	275	298.5	1104.5	33.23402	-	-	-	-
3	293	296.6667	1124.667	23.71357	-0.16549	-0.13512	0.457247472	-0.10737
4	257	286.75	2304.75	27.71732	-1.67274	-1.44864	0.142221342	-1.07039
5	228	275	5066	35.58792	-2.11961	-1.89584	0.077131518	-1.42463
6	337	285.3333	8269.333	40.66776	1.742164	1.590371	0.906520918	1.319631
7	288	285.7143	8275.429	37.13809	0.065572	0.060708	0.523028147	0.057755
8	314	289.25	8975.5	35.80802	0.761636	0.712445	0.748530168	0.669872
9	283	288.5556	9010.222	33.56006	-0.17454	-0.16456	0.436970827	-0.15865
10	411	300.8	22503.6	50.004	3.648517	3.461287	0.995724206	2.629479
11	416	311.2727	34568.18	58.79471	2.303816	2.196602	0.972175973	1.913782
12	407	319.25	42968.25	62.49964	1.628161	1.558846	0.924954116	1.439207
13	390	324.6923	47581.52	62.96925	1.132007	1.087597	0.849985287	1.03637
14	419	331.4286	52574.35	63.59387	1.497678	1.443199	0.912721246	1.357704
15	411	336.7333	55816.25	63.14171	1.251244	1.208816	0.875869425	1.154583
16	1074	382.8125	532129.7	188.3489	11.67638	11.30561	0.99999999	**5.611786**

Even before plotting a control chart, we can see a huge difference in the value of $U_{16,\,123}$, but the EWMA control charts which has been presented in Figure 2.10 shows this issue even better. Note that the standard deviation for the EWMA chart has been calculated with $\frac{\overline{MR}}{d_2}$ where the moving range is within a part among layers.

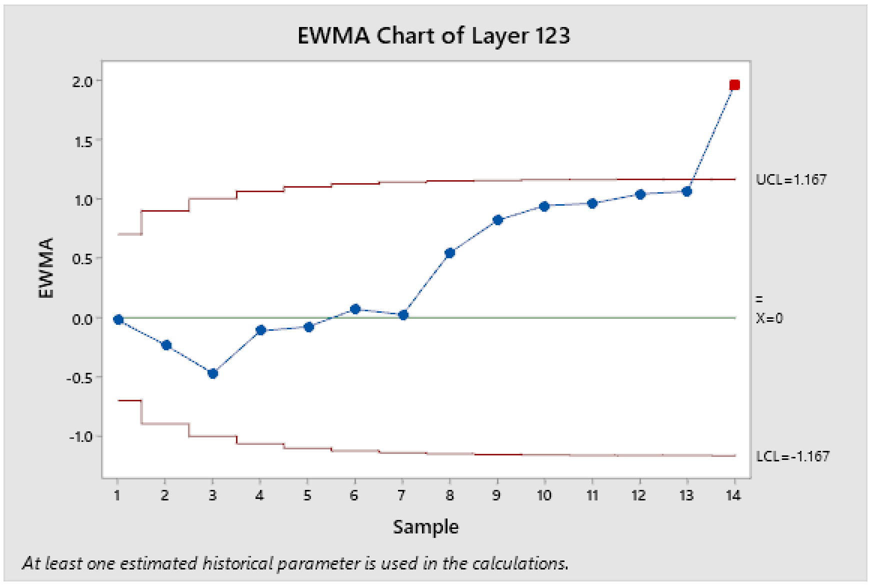


Figure 2.10. Self-start EWMA control chart for layer 123 of first 16 parts

Lightening issues and production environment changes makes some differences between same layers of different parts. Thus, trends might not have a meaning of being out of control. Also, in this method we are not looking for catching small changes in the images. So, methods like CUSUM which is basically designed to catch small shifts might not be the best method to use due to the concern of false alarms.

2.4.2. Standardizing, ARIMA filtering, change point clustering

The case study of printed parts showed sufficient performance of self-start control charts in catching shifts from images data. However, monitoring the process using this approach carries a high overhead cost since there is one control chart for each layer or in this case 243 charts). Thus, Method II can be used to monitor each part based on printed layers of same part when we printed enough parts to estimate mean and variance.

Different approaches have been explored to find optimal number of observations to design control chart. However, traditionally it is recommended at least 20-25 observation is needed to do that [64]. In Imaged base quality monitoring definition of observation might be different because there is one observation for each layer and layers from one part can be divided into different clusters. Finding sufficient number of observations to start this approach is explained better in section 2.3.2. When there are enough samples to normalizing the data from D_{ij} to e_{ij}, we need to check the independent assumption. Figure 2.11 shows autocorrelation (ACF) and partial autocorrelation (PACF) on e_{ij} of part 11 as an example which shows images (especially in first layers) are highly auto-correlated. They suggest that the underlying auto-correlated structure may be modeled by an autoregressive model.

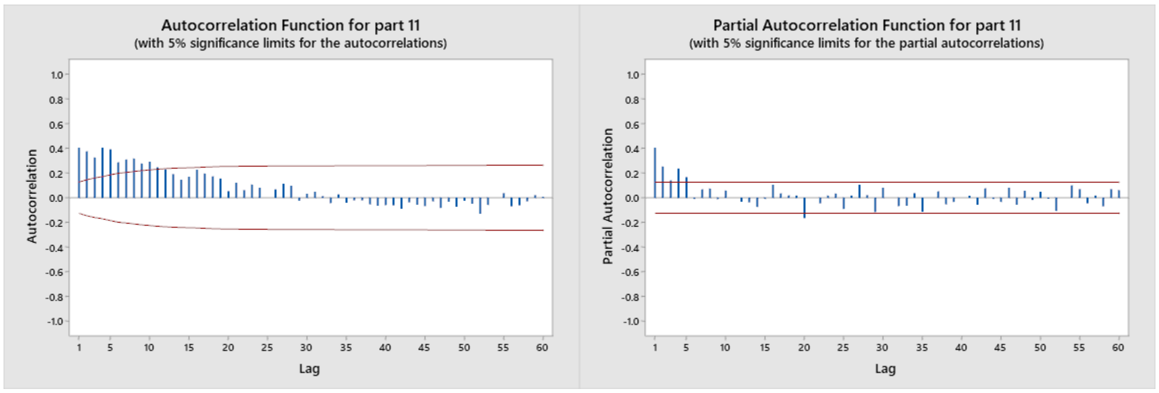


Figure 2.11. ACF and PCF of part 11

2.4.2.1. ARIMA Filter

According to the PACF and ACF in Figure 2.11 we can see there are two spikes in the PACF so we can conclude the best type of filter to remove autocorrelation would be AR(2). After applying

AR(2) over e_{ij}, the uncorrelated variable is named e'_{ij}. Figure 2.12 shows uncorrelated ACF after applying AR(2).

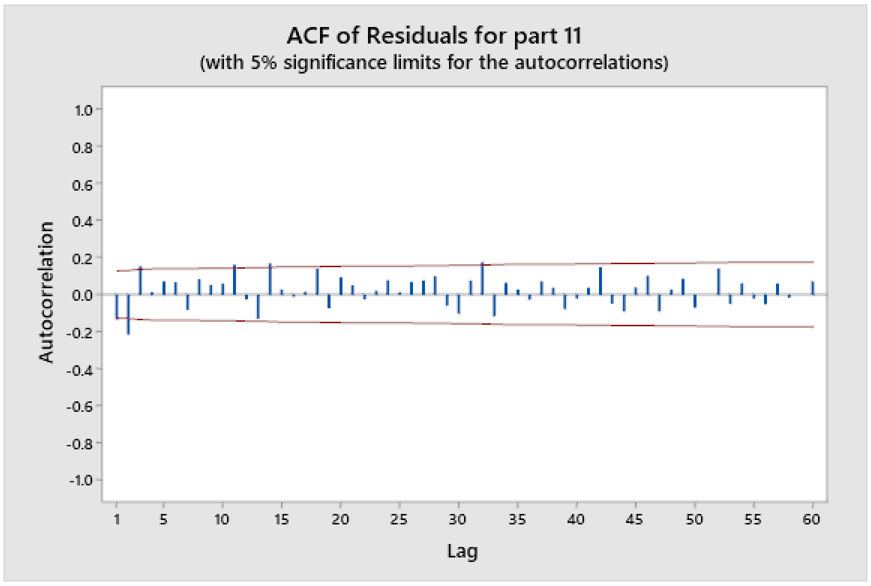


Figure 2.12. Auto-correlation (ACF) of part 11 after applying AR(2)

2.4.2.2. Change point segmentation and control charting

Next step is finding change points. To do this we can plot change point detection method proposed by Sullivan[92]for each part separately. However, in phase I of quality monitoring, we need to design our process and control limits and having different change point for different part could be a problem and is not efficient. Figure 2.13 shows time series plot of part 11and Figure 2.14 shows time series plot of location and distance for the same part. By comparing change point of all 15 parts, we conclude that for most of parts, one change happens between layer 59-63 and another between layers 185-195. Thus, the best segmentation looks to be (1, 60), (61, 180), (181, 243). Part 11 was chosen to demonstrate process monitoring with this method because of three clusters of layers. As we mentioned before, usually 20-25 observation is needed to design Shewhart control chart. In this case, since three charts are used and each chart has more than 60 observations (layers), thus 10 parts should be enough to estimate mean and variance. Thus, we can switch from the Method I self-start charting to Method II cluster-charting starting from part 11.

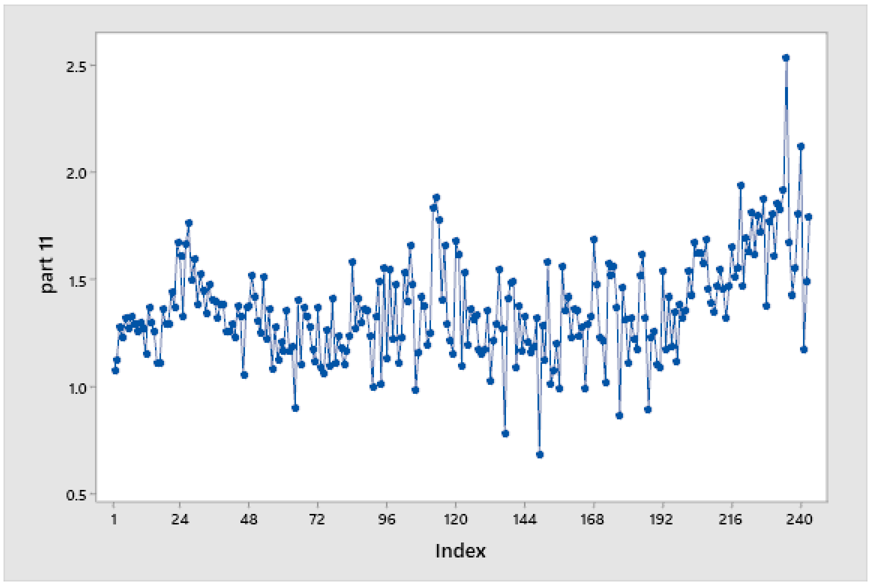


Figure 2.13. Time series plot of e'_{ij} of part 11

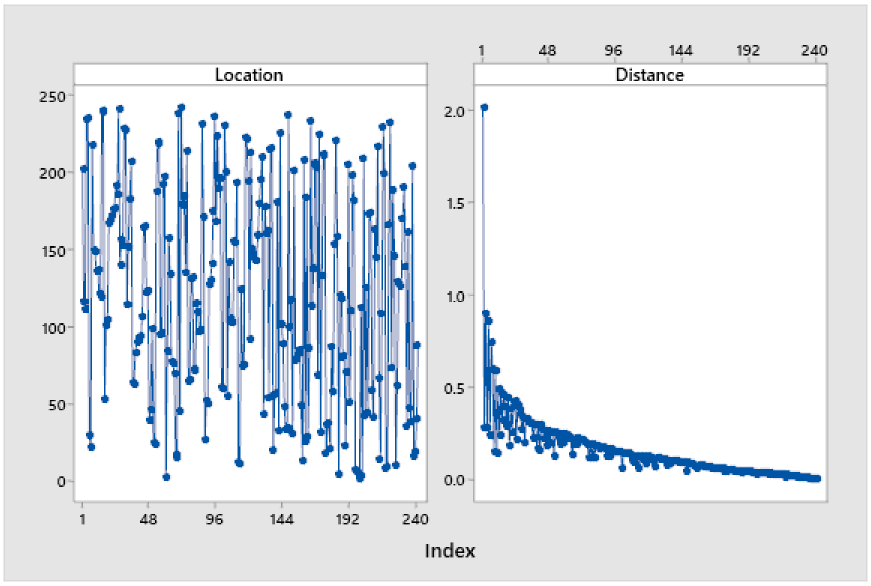


Figure 2.14. Time series plot of location, and distance for part 11

In the next (and final) step of this method, an EWMA control chart is plotted for each cluster of layers. Figure 2.15 (a), (b), and (c) shows the control chart for layers 1-60, 61-180, and 180-243 respectively.

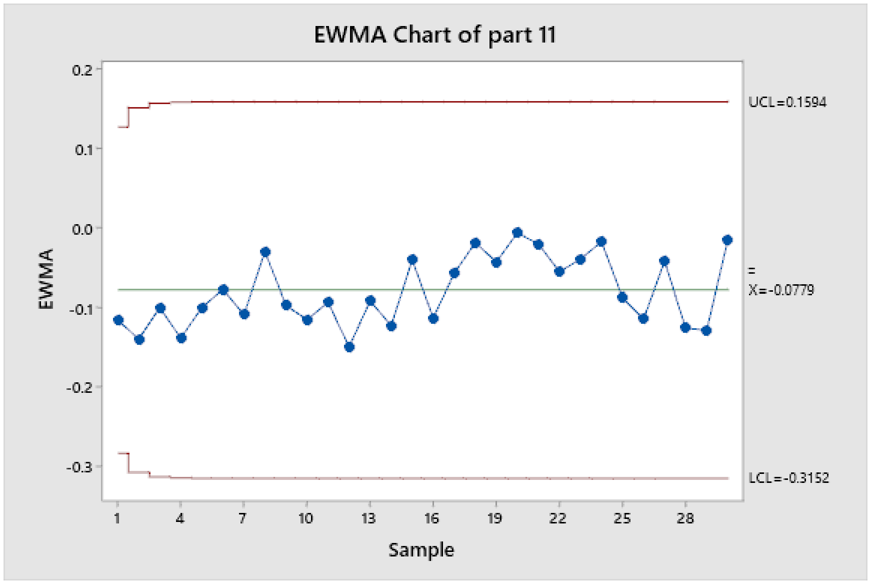


Figure 2.15. (a) EWMA control chart for layers 1-60 part 11

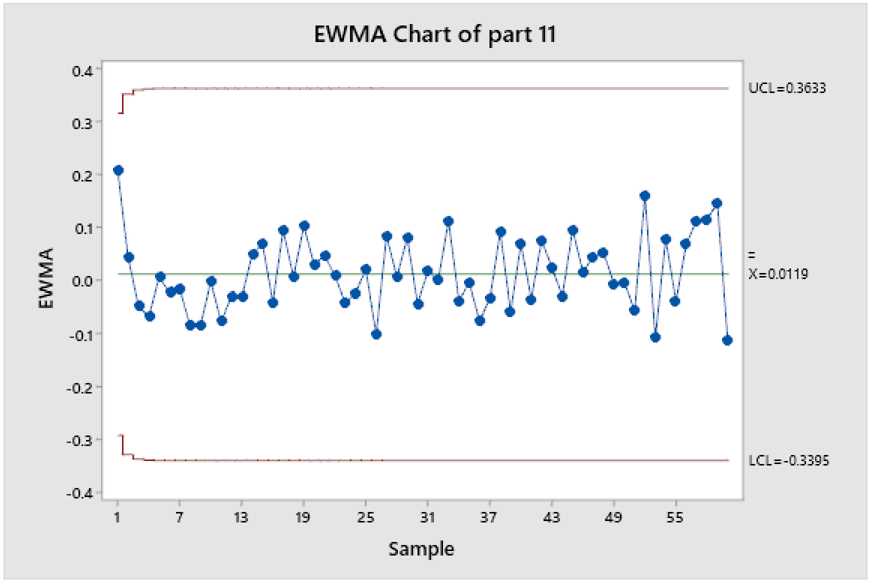


Figure 2.15. (b) EWMA control chart for layers 61-180 part 11

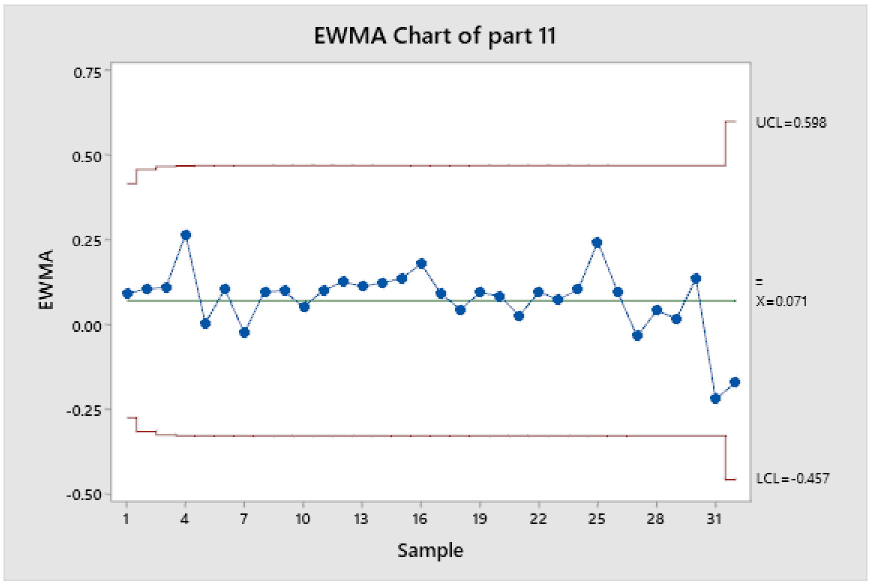


Figure 2.15. (c) EWMA control chart for layers 181-243 part 11

2.5. Conclusions and future research

This chapter presents the modeling and monitoring framework on layer-wise images in 3D printing parts. An example of 16 basket samples shows the proposed methods are implemented to demonstrate the proposed framework can indeed be implemented for process monitoring based on a very limited number of parts printed. Two complimentary methods can successfully detect printing problems layer by layer. Some important findings of this study are as follows:

- EWMA control chart can be used for imaged based quality monitoring in addition to general purpose quality monitoring.
- A self-start charting method can be used after producing only two parts. In other words, process monitoring can start from the third part. The proposed method alleviates the need of the traditional control charting phase I requirement where at least 20 to 25 parts were recommended to establish control limits.

- Images quality and lightening issue was addressed by three machine learning techniques: Neural Network (NN), Gradient Boosting Classifier (GBC) and Support Vector Machine (SVM). GBC has the best performance in terms of accuracy and false identification rates.
- After printing enough part to have an accurate estimation of mean and standard deviation of the production, Method II using the cluster charting approach may be useful to reduce the charting overhead of the self-start charting approach from 243 charts to only three charts.

In the current study, we only need to know whether the printer is producing bad part or not and there is no need to find location of the issue. For future study, this method can be extended for a problem that needs to determine the location of a problem. We will explore an approach to segment each image to smaller tiles and analyze those tiles. The problematic tile shows location of the issue as well. Also, if enough images are available and in a higher complexity situation, to determine whether a picture represents a good part or not, a deep learning and transfer learning algorithms may be used. The proposed method solves a new initial start problem in process monitoring of printing a simple part (3-inch diameter basket in our case) while deep learning may be used for printing parts with higher level of complexity. Once one deep learning module is available then transfer learning may be used to accelerate learning process with a better accuracy and less need to image data.

3. Quantifying soil, plant, and residue cover in images of agricultural fields using convolutional neural network

Abstract:

Residue cover and green canopy cover are key elements in the conservation of soil and water resources in agriculture. However, there is lack of tools to rapidly and accurately assess the percentage of the land cover covered by crop residue, live vegetation, and bare soil. This research aims to explore a deep learning model for quantifying the percentage of stubble, live vegetation, and bare soil in downward-facing images of agricultural fields. The trained model is based on a convolutional neural network (CNN) coupled with image segmentation techniques. The proposed tool based on this CNN model have features for easily and accurately quantifying green canopy cover, counting plants, and classifying stubble using a comprehensive dataset containing 3942 labeled images from real agricultural fields. This dataset includes a set of images spanning various agricultural scenarios encompassing different levels of bare soil, crop residue, and live vegetation. A hybrid unsupervised auto-labeling algorithm, combining Canopeo and Otsu's method was coded to automate initial labeling and then each labeled section has been checked and relabeled by an operator. A web-based app based on the trained CNN model is available publicly to classify different combinations of soil, plant, and stubble from agricultural field images in real time on https://soilwater.github.io/srpnet/.

3.1. Introduction

In modern agricultural systems, crop residue (i.e., stubble that remains on the soil surface after harvest) is considered a valuable input that contributes to nutrient cycling [98], promotes near-surface biological activity and soil aggregation [99]–[102], helps conserving soil water by reducing the evaporation rate [103], attenuates soil thermal fluctuations [104], [105], and constitutes a durable barrier that protects the soil surface from erosive rainfall events. Together with vegetation cover, soil residue cover minimizes soil detachment and physical dispersion due to raindrop impact, thus reducing the risk of soil erosion, soil degradation, and non-point source pollution from agricultural fields [106], [107]. Thus, to better manage soil and water resources, and to guide, implement, and assess the effectiveness of improved conservation practices, field agronomists, soil conservationists, and scientists need tools to accurately determine the proportion the soil surface covered with crop residue and live vegetation in agricultural fields. Traditional field methods to quantify the proportion of the land surface covered by stubble and actively growing vegetation have mostly relied on line transects across several parts of the field [108]–[110] and visual estimation using field guides containing reference image cards of known residue cover [111]–[113]. In recent years, the use of digital image analysis has resulted in new tools for measuring land cover components. For instance, smartphone applications like Canopeo [114] and Easy Leaf Area [115] have been used for non-destructive measurements of vegetation green canopy cover. However, most existing tools for determining land cover components are almost exclusively limited to measuring plant-related components and rarely discriminate other components of the land surface like crop residue. Part of the reason for the lack of tools for quantifying stubble from digital images is the complexity of the problem. Discriminating green canopy cover from the background can be effectively accomplished using simple thresholding techniques-based color ratios [116], [117]. On the other hand, crop stubble exhibits a wide range of colors, illumination, shapes, and level of entanglement depending on the crop type, state of residue decomposition, and residue distribution during harvest; all of which makes the segmentation of residue cover from digital images challenging. Traditional machine learning methods, such as random forests (RF) classifiers, have shown promising classification accuracy of soil, residue, and vegetation cover using a small dataset of ~200 images covering a limited number of agricultural scenarios and most images containing <50% residue cover [118]. For example, Figure 3.1 (a) and (c) represent two different scenarios and Figure 3.1 (b) and (d) are classified images of (a) and (c) respectively using RF model from [118]. Figure 3.1 (a) represent an easy to classify scenario in agricultural fields that the RF model was able to predict each pixel almost perfectly. However, Figure 3.1 (c) is a hard to classify situation where most of the picture is covered by stubble and the RF model miss-classified majority of stubble pixels as soil.

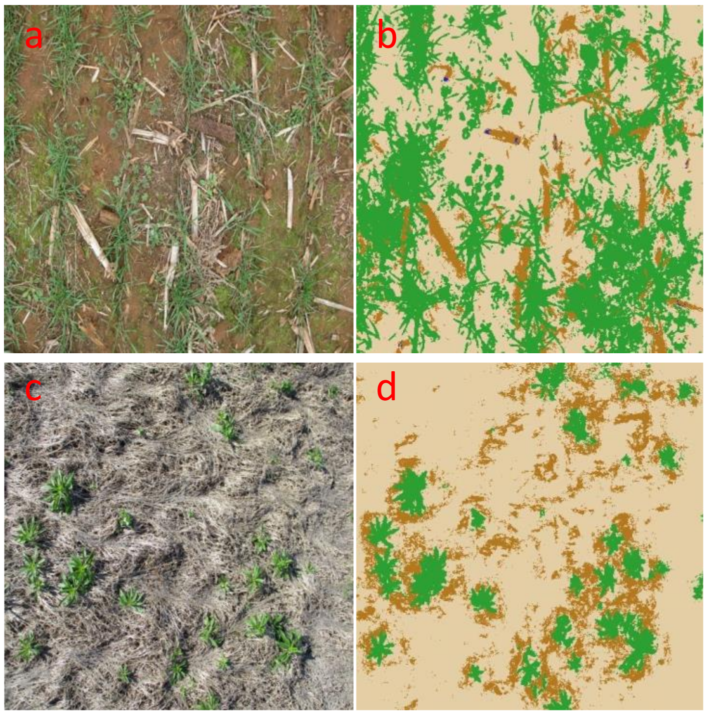


Figure 3.1. Two images from agricultural field, classified using [118] RF model. Image (a) and it’s classified image (b) represent a common simple classification scenario, while image (c) and it's classified image (d) represent another common scenario where most of the image is covered by residue and the RF model miss-classified almost all residue as soil.

Another disadvantage of traditional ML methods is that the training time is highly sensitive to the sample size, meaning that increasing the number of data points will increase computational time exponentially. For instance, the running time for a RF architecture with n training samples, k trees, and p features would result in $O(n^2pk)$ [119], which is running in quadratic time. In other words, given an input size n, the number of steps to accomplish a training task is proportional to n^2.

Convolutional neural network (CNN) methods provide a powerful framework for feature extraction and land cover classification that overcome some of the limitations of traditional machine learning approaches [120].Typically, CNNs result in better performance in terms of running time than traditional ML methods, with a complexity in the order of $O(\sum_{l=1}^{d} n_{l-1} s_l^2 n_l m_l^2)$

[54], where l is the index of a convolutional layer, d is the number of convolutional layers, n_l is the number of filters in the l-th layer, n_{l-1} is the number of input channels of the l-th layer, s_l is the spatial size of the filter, and m_l is the spatial size of the output feature map. Another distinct advantage of CNN models is their ability to take into account not only pixel colors, but also morphological and contextual information about objects. For instance, deep learning (DL) methods have been recently and successfully used for weed detection in cropped fields [121], estimation of foliar diseases in horticultural crops [122], [123] , and for root segmentation [124]. Thus, a CNN approach has the potential to learn and generalize the classification of stubble, plant, and bare soil across a wide range of complex agricultural scenarios including different crops, soils, and light conditions [125]. The objectives of this study were to i) create a benchmark dataset of semantically segmented downward-facing images obtained from agricultural fields and ii) test the accuracy of a supervised convolutional deep neural network to quantify the percentage of stubble, live vegetation, and bare soil using the generated dataset. In this study, semantic segmentation refers to the process of classifying each pixel in an image.

3.2. Materials and methods

3.2.1. Image dataset

The image dataset consisted of a collection of 3944 downward facing (i.e., nadir) pictures spanning diverse combinations of soil, plant, and stubble cover obtained from production fields and experimental plots with different tillage systems, crop stubbles, and crop phenological stages across Kansas State University Agricultural Experiment stations (Table 3.1). Images were collected using traditional point-and-shoot digital cameras and mobile devices by positioning the camera parallel to the ground at about 1.5 m above the ground level. For computational speed during training of the deep neural network model, images in the RGB (i.e., red, green, and blue) color space were resized to a width of 512 pixels and a height of 512 pixels to retain sufficient detail in the canopy structure and stubble elements.

Then, each pixel of each resized image was assigned one of three possible labels: "plant", which was defined as live, green living vegetation; "stubble", defined as crop residue left on the soil surface from previous harvests of agricultural crops; and "soil", defined as bare soil surface. Our study did not include any other objects (e.g., stones) and did not account for different plant types (e.g., crops and weeds). The labeling process consisted of a three-step approach. The first step involved a coarse segmentation of pixels corresponding to live vegetation using the approach proposed in the Canopeo application [114]. The second step consisted of segmenting the remaining pixels (i.e. non plant pixels) into stubble and bare soil using an unsupervised adaptive thresholding method based on the local mean intensity around the neighborhood of each pixel [126]. This process misclassified portions of the image, particularly regarding bare soil and stubble components, but it allowed us to generate a first-order segmentation that substantially alleviated the subsequent manual labeling of the images. The third step in the labeling process involved a manual inspection and pixel-wise correction of each image by a trained operator to ensure correct classification of the labeled image dataset. The entire labeling process was conducted using the Image Labeler application in Matlab R2020a (Mathworks, Natick, MA).

Table 3.1. Number of images in the dataset grouped by the predominant labels in the image.

Category[†]	Number of images

Soil	212
Stubble	262
Soil and plant	410
Stubble and plant	112
Soil and stubble	461
Soil, stubble, and plant	1846

† The “plant” category is missing because even images with nearly complete green canopy cover exhibited small areas with background pixels that were attributed to soil. Images with a large (>95% green canopy cover) proportion of green canopy cover were included in the “soil and plant” category.

3.2.2. Model architecture and hyper-parameters tuning

In this study we used a deep convolutional neural network (DCNN) approach called SegNet (Segmentation Network) [24] for semantic segmentation of land cover images with including a dropout after each convolutional layer as it is shown in Figure 3.2. The SegNet is an encoder-decoder architecture that includes five encoders and consists of 13 convolutional layers included in the VGG16 network [127]. Each encoder produces a featured map with the help of a convolutional layer that applies dot product by sliding a filter sized patch across the two-dimensional image. The encoder also includes a convolutional layer followed by a batch normalization layer [128] and a rectified linear unit layer (ReLU) [129]. The architecture discards the fully connected layers and only retains high-resolution feature maps in the decoder outputs, which reduces the number of trainable parameters from 134 million in VGG16 [127] to ~15 million parameters in SegNet. The last layer of the SegNet model has a Softmax classifier that outputs pixel labels based on the maximum label probability of each pixel.

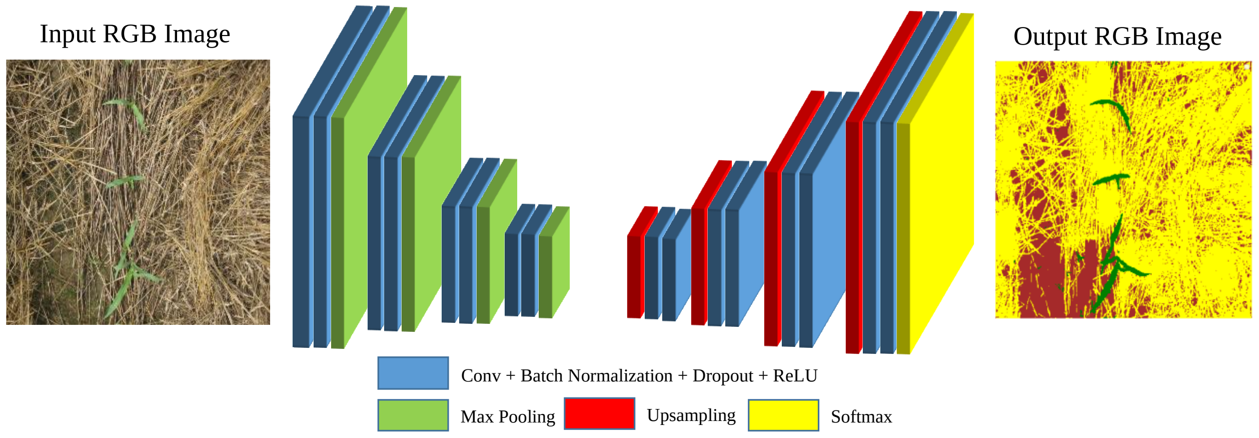


Figure 3.2. Architecture of SegNet used in this study. Each blue block representing four layers of convolution, normalization, dropout, and ReLu.

The SegNet model was implemented using Tensorflow Core version 2.1.0 in the Python programming language version 3.7 on a desktop computer with a 3.8 GHz AMD Ryzen 9 with a 12-core processor, 32 GB of RAM memory, and an NVIDIA GeForce RTX 2080 graphic card. Other DCNN architectures such as DeconvNet [130] and U-Net [131] have shown promising results for semantic segmentation in computer vision problems in medical applications [132], [133], but the retention of all the feature maps in the decoder together with the larger parameter space usually demand more computer memory than a SegNet model and result in comparable performance [24].

During the early stages of the training process to optimize the hyper-parameter space, we conducted a series of tests with the aim of fine-tuning learning rates, optimizers, batch size, and dropout parameters. Table 3.2 demonstrates the relative tuning where root means square propagation (RMSProp) and adaptive moment optimization (Adam) optimizers are two widely used optimizers in computer vision research [134]. A larger batch size might reduce each epoch running time; however, it will increase memory in use exponentially. Facing the limitations on batch size and based on available hardware, we use batch size 1 and 4. While we try different values for on parameter, the other parameters stay fix at the baseline experiments.

Table 3.2. The hyper-parameter tuning. For the process of training, hyper-parameters were tested under the key main feature of Learning rates, optimizers, batch size, and dropout. The table demonstrates the relative experiment.

Parameter	**Variations**	**Baseline experiment**	**Selected value**
Learning rates	0.1 and 0.01 with a momentum of 0.9.	0.01	0.01
Optimizers	RMS prop, Adam optimizer to identifying the better results in training the data	RMS prop	RMS prop
Batch size	1 and 4, staying under the hardware constrain the maximum constrains maximum batch size was identified as 4	4	4
Dropouts	Regularization methods have been utilized to get rid of overfitting, 20%, and 40% percentages	0.4	0.4
Epochs	The number of passes through the data is varied in this section to identify the training capabilities of the model.	200	3000

The baseline experiment was selected based on the previous research and studies in the area of DL and image segmentation. After tuning the optimizer, batch size, and dropouts value, we need to find the best number of training epochs that is showed in last column in Table 3.2. This can save time and memory as well as preventing overfitting in some cases. Based on the hyper-parameter tuning, RMS prop outperformed Adam optimizer and a learning rate of 0.01 had faster convergence than 0.1. Additionally, we observed mini-batch size of 4, 40% dropout, and 3000 epochs showed sufficient result for our problem.

3.2.3. Model evaluation

After training the SegNet model and testing the performance using test set, we can calculate confusion matrix. By having true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values in a confusion matrix, the values for OA, recall, precision, and F1-score can be calculated as follow:

$$\text{OA} = \frac{TP+TN}{TP+TN+FN+FP} \quad (1)$$

$$\text{Precision} = \frac{TP}{FP+TP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

$$\text{F-1 score} = 2 * \frac{Precision-Recall}{Precision+Recall} \quad (4)$$

3.3. Results and discussion

The proposed SegNet is trained using the best parameter that was determined in the previous section and achieved overall accuracy as high as 84%. Figure 3.3 shows accuracy and loss performance after 3000 epochs using the confusion matrix (Figure 3.4). The trained model is tested over the aforementioned test set and using confusion matrix we were able to calculate overall accuracy and F-1 score for each class shown in Table 3.3.

.

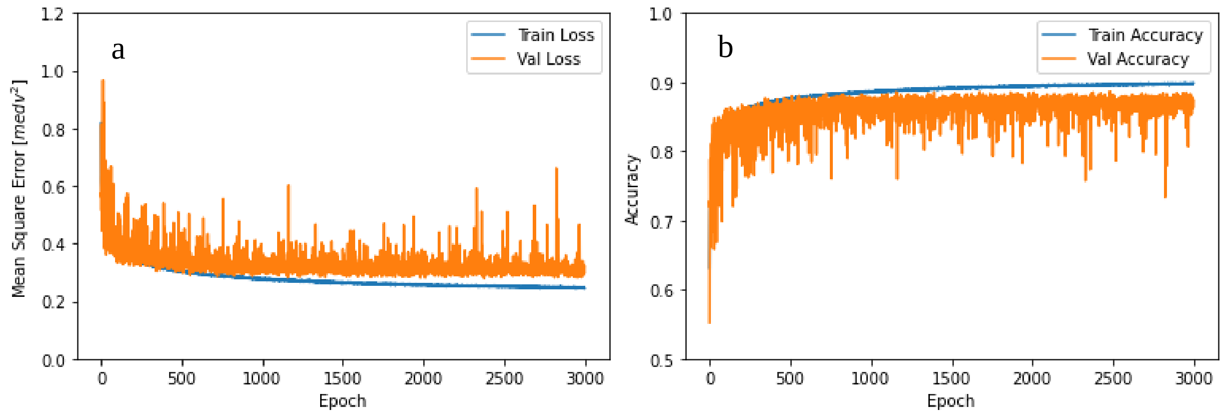


Figure 3.3. (a) Training and validation accuracy on 3000 epochs (b) training and validation loss value

	Residue	Soil	Plant
Residue	36471423	16058399	565404
Soil	5037519	69342911	1440324
Plant	1988040	2077725	35838991

Figure 3.4. Confusion matrix after 3000 epochs

From this result we can easily see that the model has no issue in classifying green canopy. However, as we expect, there is some misclassification between soil and stubble. High recall value for soil means 93 percent of pixels containing soil in our test set was detected correctly by the model. High precision value for stubble means if a single pixel is classified as stubble, then it is most likely correctly classified stubble. On the other hand, low recall value for stubble and low precision value for soil means our model is classifying some stubble as soil mistakenly. This fact can be seen better in the next section and Figure 3.5.

Table 3.3. Evaluation of trained SegNet model using a test set of 640 images

Label	Precision	Recall	F1-score
Stubble	0.86	0.69	0.77
Soil	0.79	0.93	0.85
Canopy	0.95	0.91	0.92
Overall Accuracy	-	-	0.84

Despite the complex and hard to explain nature of deep neural networks, CNNs have internal structure that preserve the spatial relationships for what the model learned during training. In this section we visualized binary classification of the final layer as well as feature maps from selected convolutional layers. Figure 3.5 shows qualitative assessment of the SegNet predictions on some agricultural RGB test images which is usually not easy to predict. Top row represents original images captured from agricultural fields. Middle Row shows how they are labeled, and the last row is our model prediction. We mention these images are hard to predict because to the best of the authors' knowledge and experience, when residue has a smooth texture (e.g., wheat residue in image E) or has a very light brown color (e.g., wheat residue in images C and D), usually ML techniques will fail to predict successfully as we show before in Figure 3.1 . This issue might happen especially when there is not enough training data.

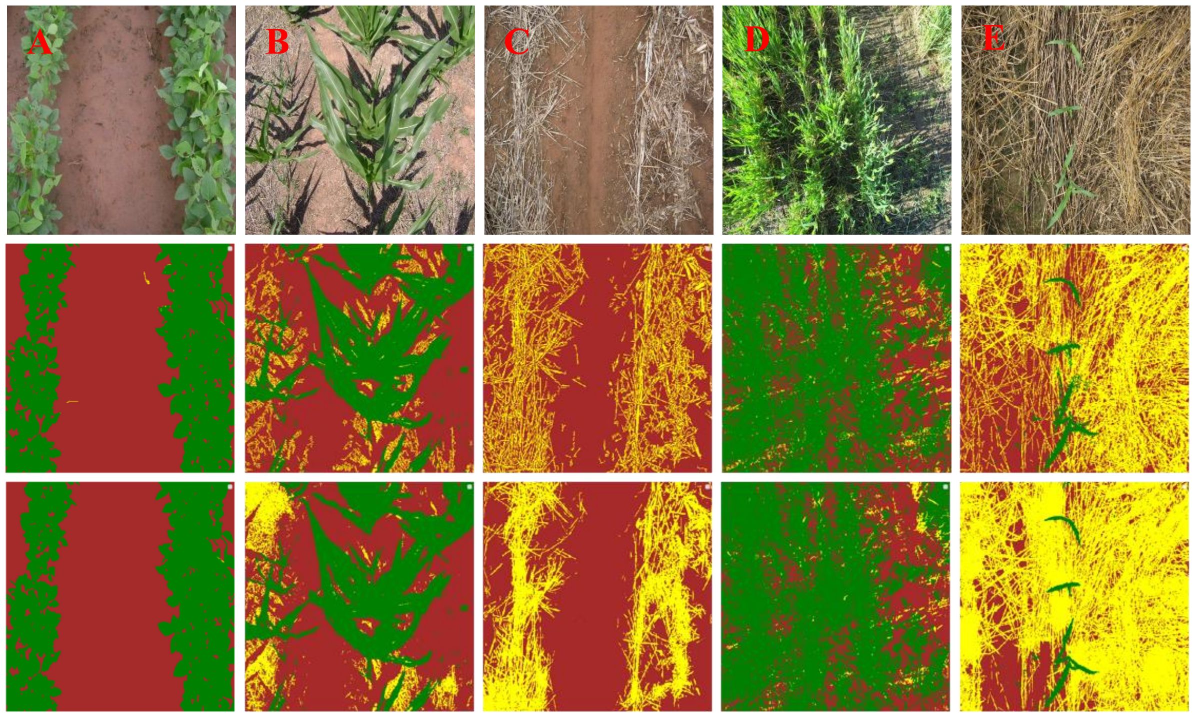


Figure 3.5. Qualitative assessment of the SegNet predictions on agricultural RGB test images. Five original images representing different scenario that might observe in agricultural field on top row. Middle row shows ground truth (labeled) images and bottom row demonstrate SegNet classification.

To better understand how the SegNet does the prediction over several hidden convolutional layers, we can have visualized feature maps after each convolutional filter. Before visualizing, we expect that the layers closer to input are detecting small details such as lines and the layers closer to output are capturing more general features representing each class of object [134]. We know our feature map matrix after the first convolutional layer has 512 x 512 x 64 where 512 x 512 is our image resolution and 64 is number of filters. This layer can be visualized by an 8 x 8 set of images shown in Figure 3.6. Since there are 4 encoders and 4 decoder convolutional layers in our SegNet architecture, we are able to visualize each and every single layer. However, because of space limitations it is not possible to include all the visualization in this paper. To bring an example of decoder layers closer to output, Figure 3.7 presents again 64 feature maps of 7th convolutional layer (or 2nd decoder layer). Note that this layer has 256 filters and visualizing all those filters is not possible in this paper. Thus, we picked the first 64 feature maps. Figure 3.6 and Figure 3.7 meet our expectations mentioned earlier in where in the previous paragraph. Feature maps in Figure 3.6 show that the model draws lines created by each object and Figure 3.7 showing the model's attempt to capture general feature maps.

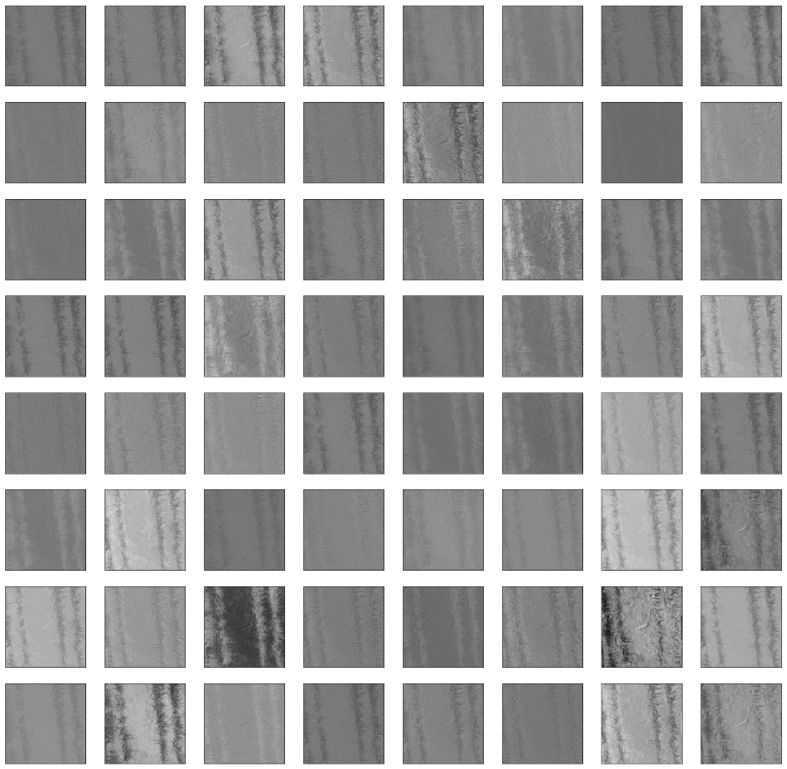


Figure 3.6. Visualization of all 64 Feature Maps Extracted from the First 16 Convolutional Layer of block 1 in the SegNet Model.

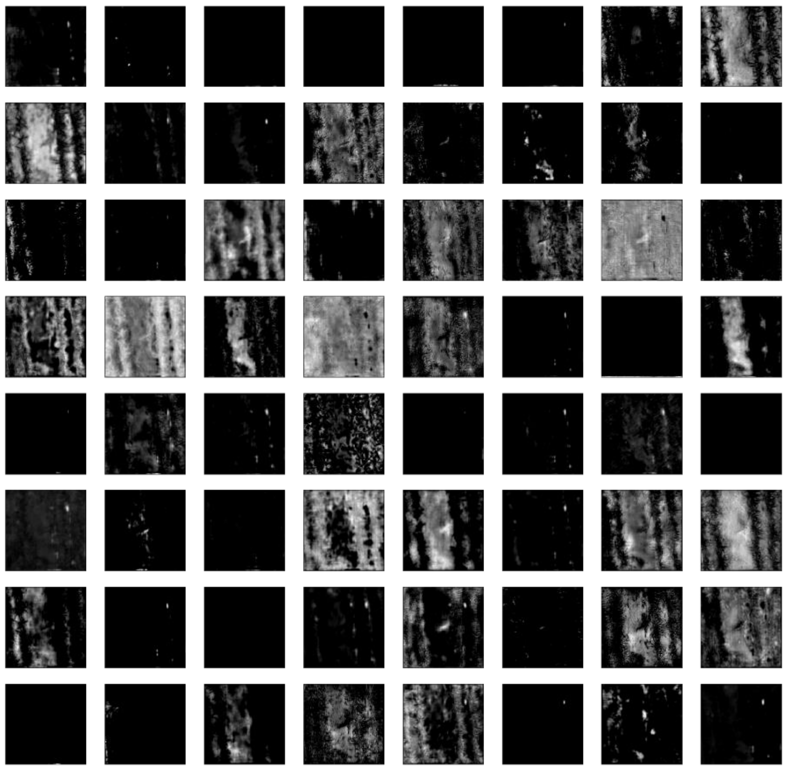


Figure 3.7. Visualization of only 64 out of 256 Feature Maps Extracted from the First 16 Convolutional Layer of block 7 in the SegNet Model

Based on the results in this section, we could conclude if the model classifies a pixel as plant or stubble, then with a high confidence it is classified correctly. However, the small issue is that not all stubble could be captured by this current model. In fact, the model misclassifies some stubble as soil. Figure 3.8 makes demonstrate of this miss-classification using the publicly available web-based application, developed by this research. Basically Figure 3.8 presents an image that is covered by only green canopy and stubble. Note that some parts are even hard for human eyes to be detected and you may need to zoom into the image to figure it out whether it is stubble or soil. The classification in Figure 3.8 shows the model successfully found all the green canopy plus all bigger pieces of stubble. However, areas with homogeneous stubble texture (shown in red areas) misclassified as soil.

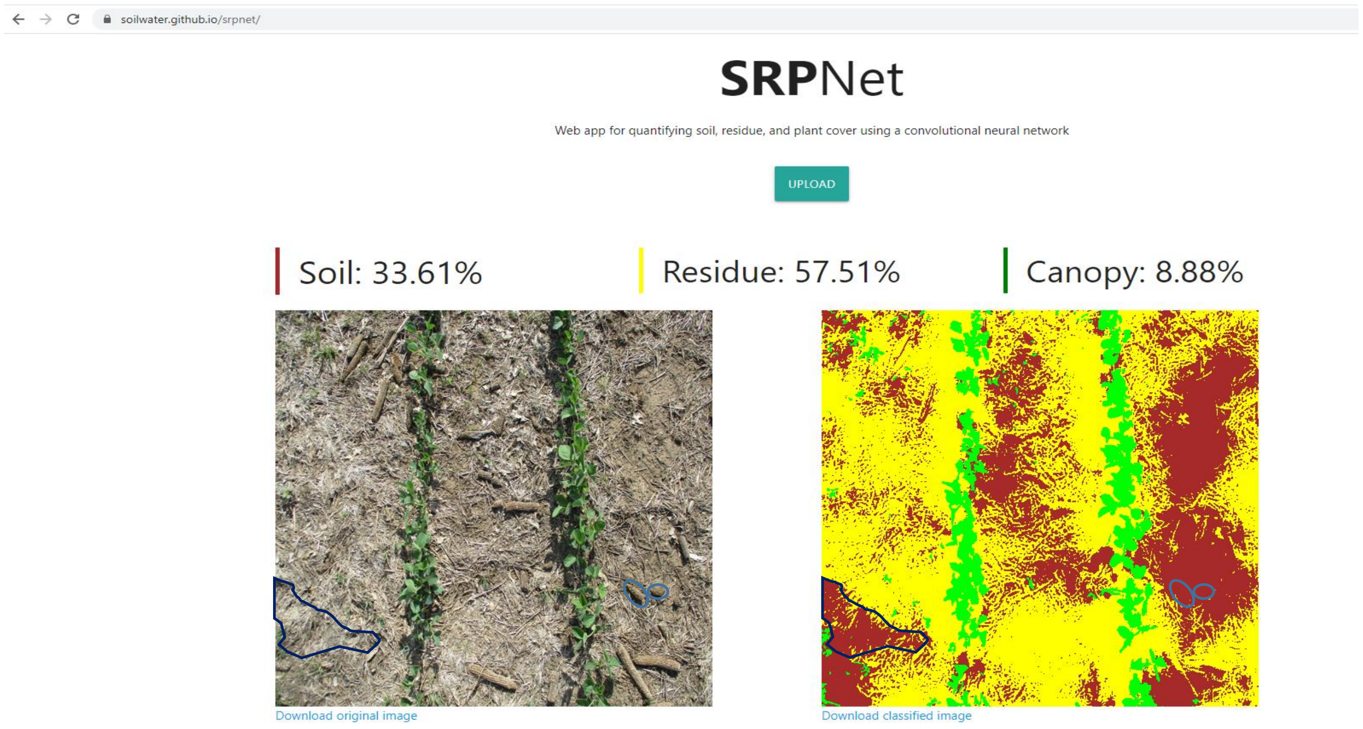

soilwater.github.io/srpnet/
SRPNet
Web app for quantifying soil, residue, and plant cover using a convolutional neural network
UPLOAD
Soil: 33.61%
Residue: 57.51%
Canopy: 8.88%
Download original image
Download classified image

Figure 3.8. An example classified image using the publicly available web-based application from this research. A scenario of the original image on the left and some misclassification from the model on the bottom left and right side of the classified image. The original image is almost covered by stubbles.

3.4. Conclusion

Residue cover is an essential component of sustainable agriculture and conservation farming techniques such as no-till farming. In this study, by providing a large dataset of high-quality images from different possible scenarios, we used a deep learning model that is able to classify soil, green canopy and stubble covered agricultural fields. Due to the tedious effort and time consumption in labeling 3942 images, a combination approach of unsupervised ML and manual correction is used to accelerate image labeling and preprocessing. The comprehensive dataset has been used to train a state-of-the-art deep learning CNN model called SegNet.

Validating the model through test sets confirms its successful performance in classifying green canopy and soil. The main challenge and future research is stubble classification where the model has an accuracy of 69 percent. The precision and recall values of soil and stubble show that the trained SegNet misclassified some stubble as soil. One simple idea to solve this issue is to retrain the model with more images from soil and stubble. Future studies could include the use of satellite images or drone videos to estimate soil cover percentage. Other ML methods such as transfer learning, semi-supervised learning and reinforcement learning could be explored to improve performance and accuracy. This research classifies three classes of soil, canopy, and stubble, but we can easily expand more classes such as stones to capture all possible varieties in agricultural fields.

4. Image-based characterization of laser scribing quality using transfer learning[2]

Abstract

Ultrafast laser scribing provides a new microscale materials processing capability. Due to the processing speed and high-quality requirement in modern industrial applications, it is important to measure and monitor quality characteristics in real time during a scribing process. Although deep learning models have been successfully applied for quality monitoring of laser welding and laser based additive manufacturing, these models require a large sample for training and a time-consuming data labelling procedure for a new application such as the laser scribing process. This chapter presents a study on image-based characterization of laser scribing quality using a deep transfer learning model for several quality characteristics such as debris, scribe width, and straightness of a scribe line. Images taken from the laser scribes on intrinsic Si wafers are examined. These images are labelled in a large and a small dataset, respectively. The large dataset includes 154 and small dataset includes 21 images. A novel transfer deep convolutional neural network (TDCNN) model is proposed to learn and assess scribe quality using the small dataset. The proposed TDCNN is able to overcome the data challenge by leveraging a convolutional neural network (CNN) model already trained for basic geometric features. Appropriate image processing techniques are provided to measure scribe width and line straightness as well as total scribe and debris area using classified images with 96 percent accuracy. Validating model's performance based on the small data set, the model trained with the large dataset has a similar accuracy of 97

[2] Reprinted with permission from "Image-based characterization of laser scribing quality using transfer learning " by Mohammad Najjartabar Bisheh, Xinya Wang, Shing I. Chang, Shuting Lei, and Jianfeng Ma, 2022. Journal of Intelligent Manufacturing, https://doi.org/10.1007/s10845-022-01926-z.

percent. The trained TDCNN model was also applied to a different scribing application. With 10 additional images to retrain the model, the model accuracy performs as well as the original model at 96 percent. Based on the proposed TDCNN classification of debris on a scribed image of straight lines, two algorithms are proposed to compute scribe width and straightness. The results show that all the three quality characteristics of debris, scribe width, and scribe straightness can be effectively measured based on a much smaller set of images than regular CNN models would require

4.1. Introduction

Laser scribing is a laser micromachining technique which uses laser scanning to make a shallow scribe line on a surface. It has been extensively studied using short pulse lasers (i.e. picosecond and nanosecond) for solar cell applications [135]–[137]. Generally, higher pulse energy and lower pulse duration mean higher productivity, but with some negative effects on scribe quality [138], [139]. On the other hand, lower pulse energy means less energy waste and melting of the process material [140]. To overcome thermally induced damage due to melting, recast and microcrack formation in laser scribing, process optimization through modeling is a viable approach.

Traditionally, scribing quality issues can be detected using optical and geometrical inspection. Variation in laser parameters such as pulse energy, pulse duration, repetition rate, and scanning speed can occur at any time scale. Those variations may result in several scribe issues such as debris, crack, missing pulse, or un-straight lines [141]. Hence, it is very important to identify and prevent these defects during a scribing process. Scribing errors are easy to be fixed right after scribing since the defect locations are known [142], [143]. Note that the usual inspection and testing for the final product (e.g. solar panels) do not help since detected defects will lead to scrapping the product.

Some recent studies deployed image analysis in monitoring different laser based manufacturing processes such as additive manufacturing [88], [90], [144]–[151] and laser welding [146], [152]–[156]. For example, Imani et al. [89] attempted to relate pore size and location to laser powder bed fusion (LPBF) parameters. In their study, they built nine titanium alloy cylinders on a commercial LPBF machine (EOS M280) at different laser power, hatching spacing, and velocity conditions. Multifractal and spectral graph analysis enabled them to monitor and discriminate process deviations with around 80% statistical fidelity. Later, Imani et. al [90] used a deep neural network (DNN) for inspection and quality control of 362 regions of interest (ROIs) representing 362 layers of a titanium alloy. A DNN algorithm called AlexNet can detect the lack of fusion flaws with 92 percent accuracy.

Various defects can occur during selective laser melting (SLM) that could be detected during the process using images (e.g. improper heat conduction in overhang features, wrong powder deposition due to a worn recoating blade, or improper heat conduction to the underneath powder at the connection between the bottom layers of the part and the supports) [146]. SLM process monitoring might be even much more challenging for difficult-to-process materials (e.g. zinc and its alloys) [148]. They compared several image segmentation methods on zinc powder ROIs to detect stable and unstable meting conditions using multivariate control charts. Their study showed that process monitoring of some difficult-to-process materials could be completely automated using suitable image segmentation techniques.

In a laser-induced material melting-solidification process, the quality of welded parts might be deteriorated by porosity, cracks, lack of fusion, and incomplete penetration [154]. Even though machine learning has been used and explored in laser welding more than in other applications of laser technology, challenges still remain in making laser welding processes more stable using advanced techniques for quality monitoring [153]. Recently Gonzalez-Val et al. [155] released first large dataset of laser metal decomposition (LMD) and laser welding. This dataset primarily includes 1.6 million images in which 24,444 of them are labeled as defect.

Based on the general performance of convolutional neural network (CNN) on image data, [153] examined a shallow CNN to monitor irregular weld seam, recessed weld seam, undercut, weld bead, and holes and spatters in laser welding. Their combined quality monitoring system was able to detect 209 out of 227 bad parts. Shevchik et al. [152] used hard X-ray radiography images to train a supervised DNN to reveal the unique signature of sub-surface events in wavelet spectrograms from the laser back-reflection and acoustic emission signals. Using 300 images in training and 100 in test set, their quality classification was able to achieve an accuracy between 71% and 99% [152]. Shevchik et al. [156] adopted a graph support vector machine with data adaptive kernel approach and 23 laser welds as the dataset to achieve an accuracy ranging between 85.9% and 99.9%.

Current physical models are capable of predicting certain geometric aspects of laser scribing such as scribe width and depth. However, several other important quality measures cannot be obtained from the model. These quality measures include heat affected zone, debris, and micro-cracks [157]. Roozbahani et al. [142] defined discontinuance as a kind of defect in laser scribing and tried to detect discontinuance area in copper indium gallium selenide solar panels using a particle analysis algorithm. However, laser scribes might also suffer from several other quality issues.

To the best of our knowledge, no existing models can predict all aspects of laser scribing quality, which can be attributed to the following three main reasons. First, laser scribing using short laser pulses is a very complicated process involving many laser parameters and various physical processes in which mechanisms are not completely understood. Second, ultrafast laser-matter interaction is a highly dynamic process and materials are first pushed to a highly non-equilibrium state followed by a rapid hydrodynamic motion, resulting in material ejection. During this process a material experiences fast phase changes and property changes (e.g. physical, optical, mechanical, electrical, etc.), making it extremely difficult to obtain reliable material data to feed into a model. Finally, the uncertainties associated with physical equipment (e.g., laser power fluctuation) and environment (e.g., temperature, vibration) can derail a model from giving reliable predictions since many of these process variations are treated as noise and thus not being considered in a physical model.

With the advent of machine vision and machine learning (ML), an opportunity arises for an Artificial Intelligence (AI) framework to be used to monitor and characterize a laser scribing process with multiple quality features including debris, scribe width, and scribe straightness. The inputs of the proposed framework are images while the scribing is taking place and the output is a classification reports on the quality characteristics under consideration. We propose a deep learning method for the AI framework. Considering the fact that each problem might need a new dataset and high cost of providing labeled dataset for supervised ML that give sufficient accuracy, the main challenge is to use the least amount of images possible to train such a model and able to monitor all aspects of aforementioned scribing quality.

Deep learning (DL) and convolutional neural network (CNN) showed promising performance among other ML methods in recent years in different contexts from autonomous driving to medical image analysis [24], [25]. However, these methods need significant amount of data for training a model with adequate accuracy [26]–[28]. Collecting image data may not be a big challenge, but pre-processing and labeling of these images for training is. This data preparation stage is the most time-consuming and costly step in any machine vision/DL applications. One way to alleviate this problem is the use of transfer learning [29].

Existing supervised ML methods such as decision tree and other methods based on various trees such as Random Forest (RF) or Gradient Boosting Classifier (GBC) may not need as much training data as the DL/CNN models would but still require a large amount of data. In addition, these traditional ML methods may not be able to handle complicated problems such as semantic segmentation with multiple quality characteristics. In such a complicated problem, there might be several classes of objects to be classified or each might have different geometrical shape and color [28]. A new approach to alleviating the lack of labeled data problem is called Transfer Learning (TL) where the knowledge gained from a different and yet similar problem can be used to solve another problem. Figure 4.1 demonstrates how the knowledge could be transferred from a pre-trained model to a new model where Softmax layer is an extension version of logistics regression idea into a multi-class world.

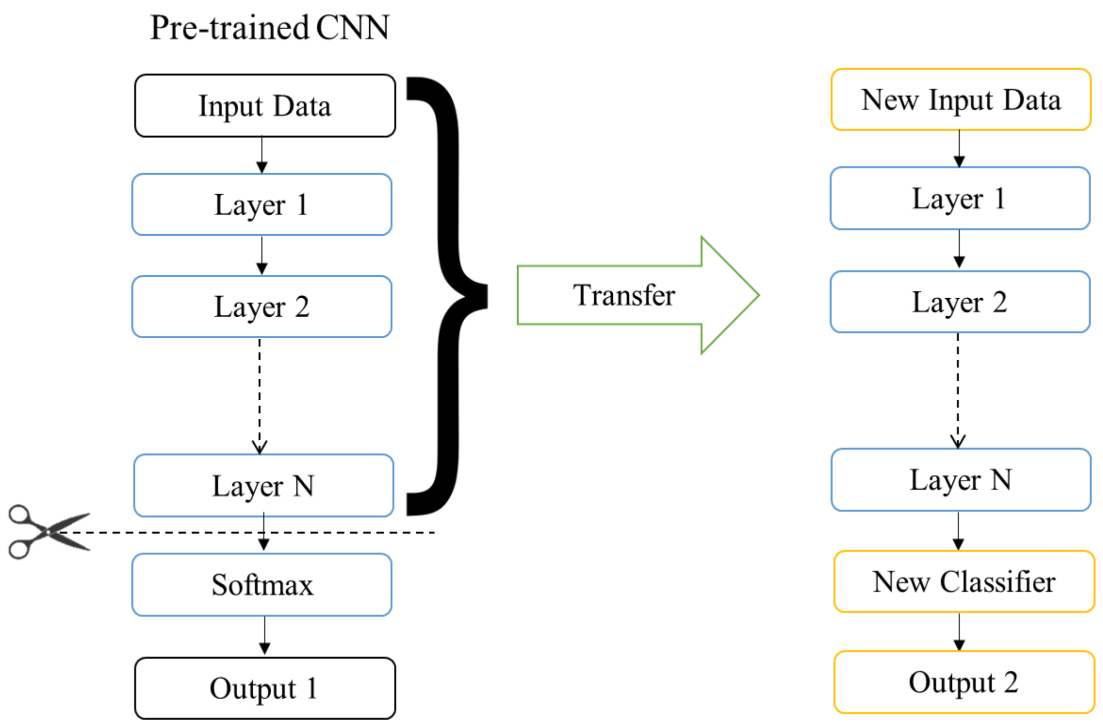


Figure 4.1. An example of feature transfer from a pre-trained model in TL

In the last few years, several studies used image data to monitor laser-based manufacturing processes. Table 4.1 summarizes these studies, the research focuses, and their results. However, there is a lack of research for in-process monitoring of laser scribing quality. These quality characteristics include scribe width, debris, crack length, scribe depth, width of heat affected zone, and straightness. Also, most of the research is done using a specific experimental condition and one question is whether the results could be applied to other scribing conditions with different imaging systems. In this study we attempt to measure and monitor three important laser scribing characteristics (i.e. scribe width, debris, and straightness) using image data and a state-of-the-art transfer learning method. TDCNN will enable us to leverage existing deep learning models from different domains and accelerate classification with less amount of laser-scribing data.

Table 4.1. Applications and features addressed by the most related publications

	Study	*Year*	*Case*	*ML/Monitoring approach*	*Semantic segmentation*	*Debris*	*Scribe width*	*Scribe straightness*	*Other quality characteristics*
1	Grasso et al. [158]	2017	SLM	principal component analysis (PCA) and k-means clustering	No	No	No	No	local overheating phenomena caused by a wrong heat transfer from the melt pool to the surrounding material.
2	Grasso et al. [148]	2018	SLM	IsoData, Otsu's, Li's, Huang's and k-means, and multivariate control-charting	No	No	No	No	detect unstable melting conditions since their early stage.
3	Imani et al. [147]	2018	LPBF	multifractal & spectral graph theoretic analysis	No	No	No	No	size, count, and location of pores around 80% accuracy and images from nine samples.
4	Imani et al. [90]	2019	LPBF	DNN (AlexNet)	No	No	No	No	lack of fusion flaws; 362 images from one single part reach accuracy 92.5%.
5	Roozbahani et al. [142]	2018	Laser scribing	A Particle Analysis Algorithm	No	No	No	No	Detecting discontinuance area in copper indium gallium selenide solar panels
6	Mayr et al. [153]	2018	Laser welding	CNN	No	No	No	No	irregular weld seam, recessed weld seam, undercut, weld bead, and holes & spatters. 500 operations (images) with 512x512 resolution.
7	Shevchik et al. [156]	2019	Laser welding	graph support vector machine with data adaptive kernel	No	No	No	No	23 laser welds for the same events in row 9.
8	Shevchik et al. [152]	2020	Laser welding	DNN	No	No	No	No	300 images in training and 100 in test set is used to reveal the unique signature of sub-surface events in wavelet spectrograms from the laser back-reflection and acoustic emission signals.
9	Zhang et al.[154]	2020	Laser welding	DNN	No	No	No	No	Porosity monitoring using grayscale images and achieving 96.1% accuracy.
10	Gonzalez-Val et al. [155]	2020	Laser welding	CNN	No	No	No	No	The model were able to detect defect parts with 97.5% accuracy.
11	**This study**	**2021**	**Laser scribing**	**TDCNN**	**Yes**	**Yes**	**Yes**	**Yes**	**The proposed model achieved 96% accuracy using a small set and 97% on large set.**

4.2. Research methodology

4.2.1. Experimental setup

The experimental setup is depicted in Figure 4.2, and the samples used throughout the study are <100>-oriented, 1-mm thick intrinsic Si wafers with a resistivity of >200 Ω cm. An IR laser is used to scribe lines on the surface of a silicon wafer. Specifically, the laser source (MWTech, PFL-1550) has a wavelength of λ = 1550 nm and produces pulses of length τ = 3.5 ns (full width at half-maximum) and can be operated at various repetition rates with a maximum pulse energy of 20 µJ. The output beam has a 1/e2 diameter of 6 mm.

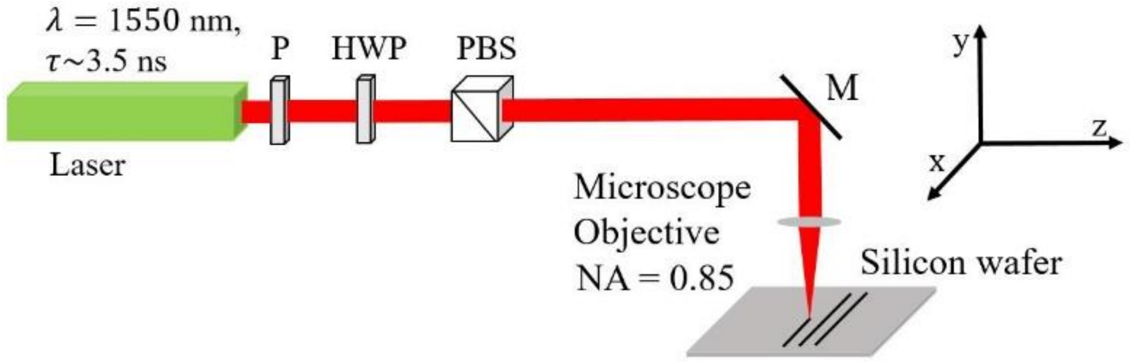


Figure 4.2. Experimental setup. P: polarizer, HWP: Halfwave plate, PBS: polarized beam splitter, M: mirror.

A half wave-plate in conjunction with a polarizing beam splitter is used to control the pulse energy by rotation of the wave-plate. The beam is then focused on to the surface of the Si sample by a microscope objective (NA = 0.85, Olympus, Model LCPLN100XIR) that is corrected for spherical aberration. At focus, the beam has a theoretical diameter at $1/e^2$ of $2w_0 = 1.22\,\lambda/NA = 2.2$ µm, with a Rayleigh length of y_R = 2.6 µm in air. Parallel lines are scribed on the surface of the silicon samples. Considering three parameters in control and easy to change, we adopted a 23 factorial experimental design. Each factor is experimented on a high and a low value.

Specifically, the low level and high level of the pulse energy are 1 and 2 μJ respectively. The low and high levels for petition rates are 20 and 120 kHz. Finally, the low and high setting for scanned speed are 0.5 and 10 mm/s respectively. The scribing conditions are listed in Table 4.2. The images are obtained by separating long scribe lines from each condition listed in Table 4.2 into multiple small segments.

Table 4.2. Scribing conditions used in our experiment

Line number	*Pulse energy (μJ)*	*Repetition rate (kHz)*	*Scanning speed (mm/s)*
1	1	20	0.5
2	1	20	10
3	1	120	0.5
4	1	120	10
5	2	20	0.5
6	2	20	10
7	2	120	0.5
8	2	120	10

4.2.2. Image data and pre-processing

The first step of image processing for object identification is image segmentation. This segmentation task is accomplished by an unsupervised ML model, which does not require a time-consuming labeling process. However, unsupervised ML has limited applications and are not suitable for problems that need high accuracy. Figure 4.3 (b) shows adaptive thresholding (AT) [159] and Figure 4.3 (c) Otsu's thresholding (OT) [160] methods, respectively. AT is a local intensity method while OT is a global intensity one. As shown in Figure 4.3, AT performs better than OT. However, the segmentation is still very far from desirable for process monitoring. To monitor a process, a high level of segmentation accuracy is required. Neither method achieves this standard. Our goal is to measure and monitor scribe width, debris, and straightness. None of those goals can be achieved using these thresholding methods.

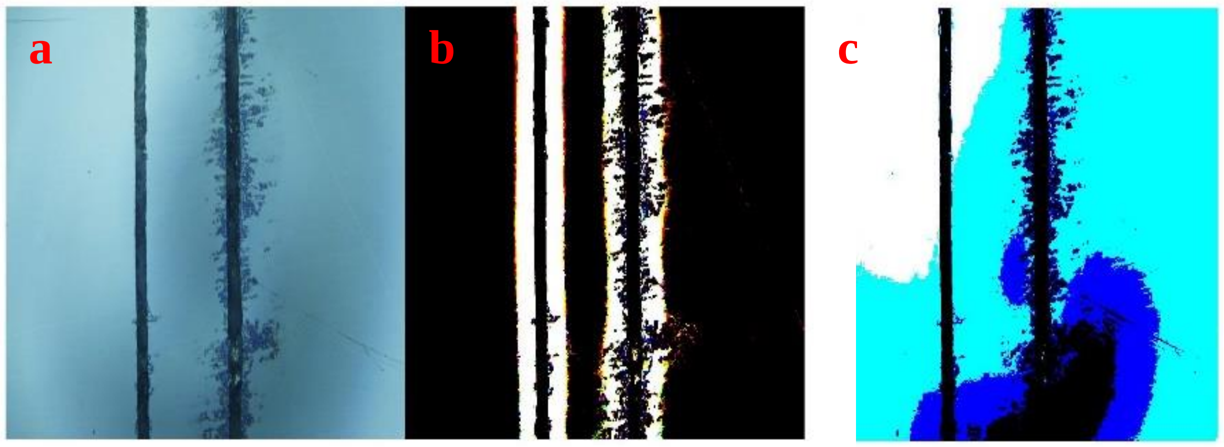


Figure 4.3.(a) A sample include 2 scribes, (b) clustering result using adaptive thresholding (AT) method, (c) clustering results using Otsu's thresholding (OT) method

Image pre-processing can include renaming, resizing, de-noising, segmenting, edge smoothing, and finally labeling. In this study, the collected image dataset is renamed, resized to 1024 x 1024, and labeled to the mentioned classes of scribe, debris, and the part background. The initial goal is to train the model with sufficient accuracy and minimum amount of data. To do this, a total of 21 images from 8 different scribes are collected and labeled to three classes of debris, scribe, and silicon background. Note that this data sets were split into 3 sets of training, validation, and testing with the ratio of 60-20-20. However, a valid concern is testing the model is not reliable based on a handful of images, even if we get very high accuracy and low loss. To make sure the accuracy is reliable, we prepared a large dataset that includes 154 images with the same size and more variety in scribe size, camera zoom, and defects. The purpose of the second dataset is to verify model performance. Specifically, we define the validation set as the dataset held back from training to estimate the model's capability for tuning hyper parameters and the testing set is just as the part of dataset held back from the training set to give an unbiased estimation of the final trained model [161]. However, by verification we want to ensure that the model won’t misbehave on a broader range of circumstances [162]. Thus, in this research, all the images were labeled carefully using MATLAB R2020a Image Labeler application manually. A pixel-wise region of interest is defined in the MATLAB Image Labeler application where we assigned 1 to all silicon background pixels, 2 to all scribe pixels, and 3 to all debris pixels. The labeled ground truth data was exported from MATLAB environment and then imported to Python for DL and image processing.

4.2.3. Transfer learning model architecture

To solve traditional machine learning issues pointed out in the previous section in image segmentation, we designed our TL-based model. Figure 4.4 shows the details of the designed architecture where blue part (i.e. the first two and half rows) represents the layers with weights transferred from the pre-trained VGG16 and the orange part (the rest of the rows) is the proposed CNN classifier built on top of the pre-trained model. TL works the best when a related pre-trained CNN can be used to transfer knowledge. There are several well-known pre-trained CNNs such as Xception, VGG16, VGG19, ResNet50, Inception, and MobileNet, which have been trained over different public datasets like ImageNet and MNIST, and CIFAR [130], [163]–[166]. All of these datasets are designed for object detection, of which goal is just to determine whether an image contain a specific object or not. However, to the best of our knowledge we could not find any pre-trained pixel-wise semantic segmentation CNN. Among several aforementioned pre-trained models for image classification, VGG16 is a very deep CNN that is trained on part of ImageNet dataset with 2 million images and 1000 different class of objects such as animals, furniture, sports, plants, etc. [167]. VGG16 has a high accuracy for the objects it was trained for. Thus, for this research, VGG16 is chosen to transfer the image feature knowledge.

Note that the original VGG16 is trained on images with a 224 x 224 resolution. The output of this model is one scalar value representing the classified object. However, the input, desired output, and consequently dimensions of all the layers need to be changed for different problems. In each problem, various resolutions for images can be used for training. Thus, the main adjustment needed for the proposed framework is to change the input dimension or image resolution. The topology of the proposed TDCNN is shown in Figure 4.4. Working on appropriate fine-tuning and feature extraction, we trained the proposed TDCNN model. This new classifier can be a logistics regression or support vector machine model in case of binary classification, or deep CNN. Since the pre-trained model and the proposed model serve two purposes (the former is for object detection and the latter is for semantic segmentation), thus another deep CNN should be trained based on a new small dataset. To design this deep CNN, the idea of decoding and up-sampling in Unet [131] and SegNet [24] is adopted. In a similar study, pre-trained DeconvNet [168] layers are transferred on top of VGG 16 for off-road autonomous driving without considering any batch normalization or drop outs [169].

The last blue block in Figure 4.4 is the output from the feature extraction process. The output of feature extraction from the pre-trained model is a matrix with dimensions of 128x128x256 where 256 is the number of filters. This output (i.e. the output from the last blue block) is the input of the proposed CNN. Six blocks of convolutional layers are designed. Each block contains a convolutional layer, batch normalization, ReLU, dropout, and Max pooling (or unpooling). The dropout helps to reduce overfitting. There is only one max pooling layer connected right after the first convolutional block followed by four max unpooling layers for up-sampling weights to the desired dimension, which is 1024 x 1024 in this case. Finally, a fully connected layer followed by a Softmax layer gives the desired classification. Note that the weights in the blue part remain fixed during training. This is because those weights have been obtained with training from millions of images. The rationale of transfer learning is to leverage this knowledge of fundamental geometrical features for various objects. The orange part is the proposed module for the desired classification of a specific problem domain, in this case, the scribed images.

Despite the complex appearance of the proposed architecture, it is significantly less complicated than the other well-known CNNs like the VGG16. As mentioned above, the blue section in the architecture belongs to the transferred layers from the pre-trained VGG16. The blue part does not include all the trainable layers in the original VGG16. The weights in the transferred layers from VGG16 were unchanged during training. In the proposed TDCNN and based on filter size and number of layers, we only trained 5 million parameters in each epoch for the new module with 771 parameters in the last dense layer, which were 3 times less than those of VGG16.

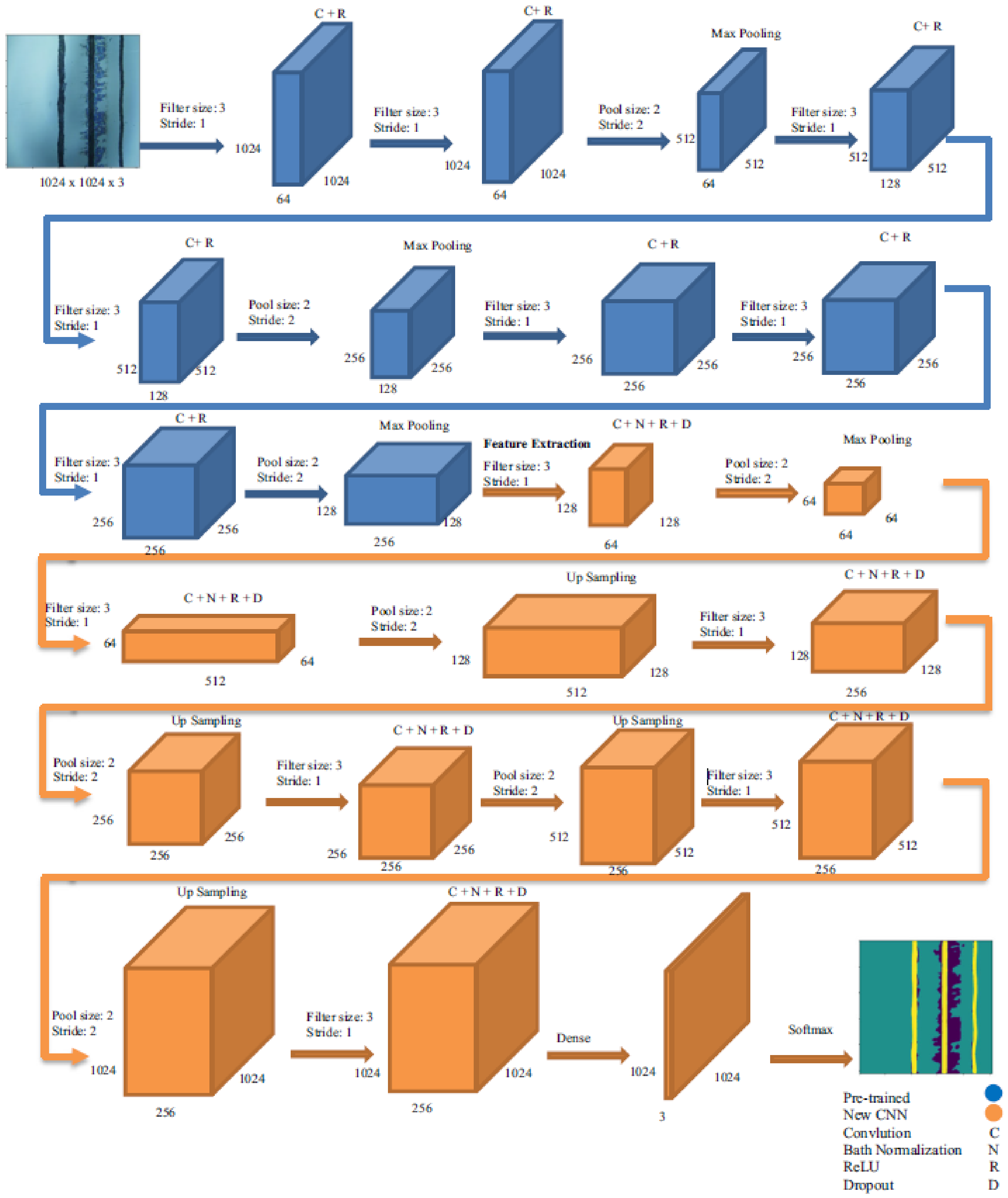


Figure 4.4. The proposed TDCNN architecture

4.2.4. Model training and evaluation

The architecture in Figure 4.4 was coded on the Google Colaboratory using a P100 GPU and TensorFlow environment. Model evaluation was done by two sets of image data; a validation set and a testing set. The validation set was the part of the sample data held back from training to estimate model performance during tuning the model's hyper-parameters while the testing set was the part of the sample data held back from training to estimate the model's final performance. We split the entire small dataset with 21 images to 14 images for training, 3 for validation, and 3 for testing. It is necessary to emphasize that the main effort here is to train our model with a few numbers of images and reach desired accuracy. Since the training set was small, the test set was also small, and one might claim the result was not enough. To address this concern, a large data set including 154 images was created and trained later to verify the results and accuracy.

In the training phase, training and validation accuracy and losses were used to access model performance. Thus for the aforementioned model, using Adam optimizer [170] with a learning rate of 0.0001, and a mini batch size of 4 the model was trained for 30 epochs. Figure 4.5 demonstrates the training performance of the designed model. Note that a dropout value of 0.4 was used to prevent possible overfitting during training. Both training and validation indexes (i.e., accuracy and losses) in Figure 4.5 are very close to each other in each epoch, especially closer to the final epochs. This observation demonstrates that our model was able to avoid overfitting successfully.

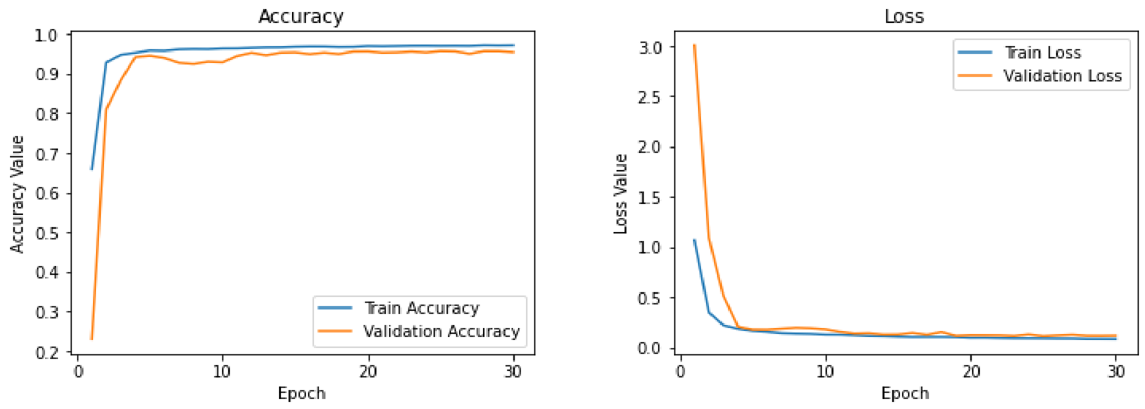


Figure 4.5. Training and validation performance of TDCNN

While Figure 4.5 represents training and validation performance during the model training process, there is still a need to examine the final trained model accuracy on test set. To do this we calculated F1-score for each class and Overall Accuracy (OA) using confusion matrix. Given a general structure of confusion matrix with True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) values in it, F1-score is defined as harmonic average of precision and recall. Equations 1-4 give some details on how to calculate OA, precision, recall, and F1-score. OA is the correct identification rate. Precision is the rate of correct positive observations out of all observations identified as positive. High precision rates mean low false positive rates. Recall, on the other hand, is the ratio of correctly predicted positive observations out of all positive cases. High recall rates mean low false-negative cases. Finally, F1 score is the weighted average of precision and recall. In short, precision is a measure of false positives while recall is a measure of false negatives. F1 score is an overall measure of both. OA is the F1 score when all correct identifications of all categories are considered as a whole. All 4 metrics used here are the larger the better.

$$\text{OA} = \frac{TP+TN}{TP+TN+FN+FP} \tag{1}$$

Precision = $\frac{TP}{FP+TP}$ (2)

Recall = $\frac{TP}{TP+FN}$ (3)

F1 score = 2 * $\frac{Precision*Recall}{Precision+Recall}$ (4)

Table 4.3 shows the OA and details of F-1 scores over the test set. The last column in Table 4.3 is the number of pixels that support each class. Precision here means, for example, 64 percent of those pixels classified as debris are actually debris. This outcome is expected because debris pixels are only 6.7% of all pixel examined and there are many none debris pixels mistakenly identified as debris. The recall value for debris means our model was able to find and classify 85 percent of all debris pixels. The recall for "Part background" is over 97 percent while the recall for scribed part is 94 percent. This means that the proposed model is able to find and classify all the pixels related to the part correctly with very high rates. This outcome also suggests that most of misclassifications are related to debris and scribe pixels. Debris has the lowest F-1 score. The fact of low precision and high recall values for the debris means the model was able to find most of the debris; however, there are also many non-debris pixels that were classified as debris. We suspect that the misclassification is from the scribe. Figure 4.7 provides the evidence since many debris are connected to the scribed line.

Table 4.3. Overall accuracy and F1-score of testing the model using the first data set

	precision	recall	F1-n score	support
Debris	0.64	0.85	0.73	281939
Part background	0.99	0.97	0.98	3574063
Scribe	0.95	0.93	0.94	338302
OA (Overall Accuracy)			0.96	4194304

Since using 3 images in the test set (despite the high image resolution) might not be very reliable, we trained this model with 60 percent of images in the large dataset and the rest of them is used for testing and validation. Interestingly, this time accuracy even improved to 97 percent and loss value went below 0.1 for both training and validation. Table 4.4 shows the results of training the model and testing it. As can be seen, performance is just slightly better than training with the small set. The F-1 score for debris is still less than 80 percent. This is probably because of the high unbalance ratio. Also, considering Figure 4.5 and the same graph in training the large dataset, there is not a big gap between the training and validation loss value. This observation confirms that our model was able to avoid overfitting.

Table 4.4. Overall accuracy and F1-score of testing the model using the large dataset

	precision	*recall*	*F1-score*	*support*
Debris	0.69	0.83	0.79	663595
Part background	0.98	0.99	0.98	17048192
Scribe	0.96	0.98	0.97	3259733
OA (Overall Accuracy)			0.97	20971520

4.2.5. Generation of scribe width and straightness of laser scribes

Given a sample scribe in Figure 4.6, we can measure debris as well as scribe width and straightness. Trained models over the small and large sets had almost the same performance. In this section we use the model trained by the small set. The output from TDCNN is a 2D image that three categories of scribe, debris, and sample part are classified in it (see Figure 4.7 (b)). Thus, debris can be measured directly by measuring the number of pixels that classified as debris. Using the same classified image, scribe width and straightness also could be measured using geometric dimensioning & tolerancing (GD&T) [171], [172] with appropriate adjustments on the formulas based on the classified images.

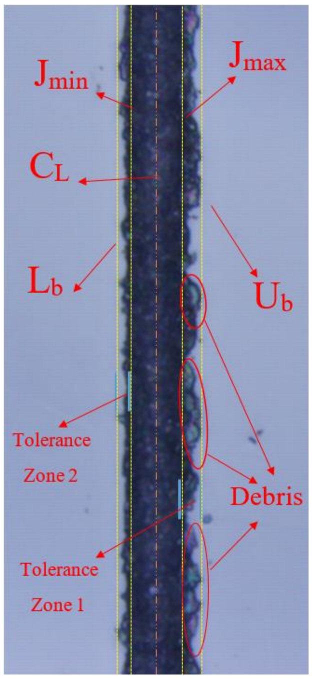


Figure 4.6. A scribed sample part showing measurement parameters

After quantifying debris, the whole image will be segmented to scribe and no-scribe pixels. Figure 4.7 shows the image and pixel values for an example image of 3 scribe lines. In the initial classification in Figure 4.7 (b), the model assigns one's to all scribed pixels, zero's to all background pixel and two's to all debris. After measuring debris area and monitoring scribe width and straightness, all the debris pixels' value will be replaced with zero's to have a binary classification of scribe and no-scribe.

In Figure 4.7 (c) a pixel matrix is presented, in which one's (1's) represent all pixels that are classified as scribed and zero's represent the background and debris. Let x_{ij} be the value of a pixel in row i and column j. In order to determine the width and straightness, the tolerance zones and center line CL need to be identified. The width (W) is the distance between Jmax and Jmin which are the maximum and minimum positions in the effective area, respectively. The effective area consists of all column j's where the row sum of j= $\sum_{i=0}^{1023} x_{ij}$ is greater than 1024*(1-α). Here α is the classification error and 1024 is the image resolution. Thus the effective area's boundary and width are:

$J_{min} \leq$ Effective area $\leq J_{max}$ (5)

And,

$W = J_{max} - J_{min}$ (6)

Let J be all the columns that with value 1 (i.e. contain scribe). Then the center line (C_L) will be:

$C_L = \sum_{j_{min}}^{j_{max}} j / J$ (7)

Now define upper bound (U_b) max j that contains at least 1 pixel contain scribe and lower bound (L_b) min j that contains at least 1 pixel contain scribe. Thus:

Tolerance Zone 1 is between line U_b and J_{max} (8)

Tolerance Zone 2 is between line J_{min} and L_b (9)

Finally, if the total number of columns in the tolerance zone is n then for each row, the error (e_i) in the tolerance zone is:

$e_i = \sum_{all\ j\ in\ tolerance\ zone} |x_{ij} - \bar{x}_{ij}| / n$ (10)

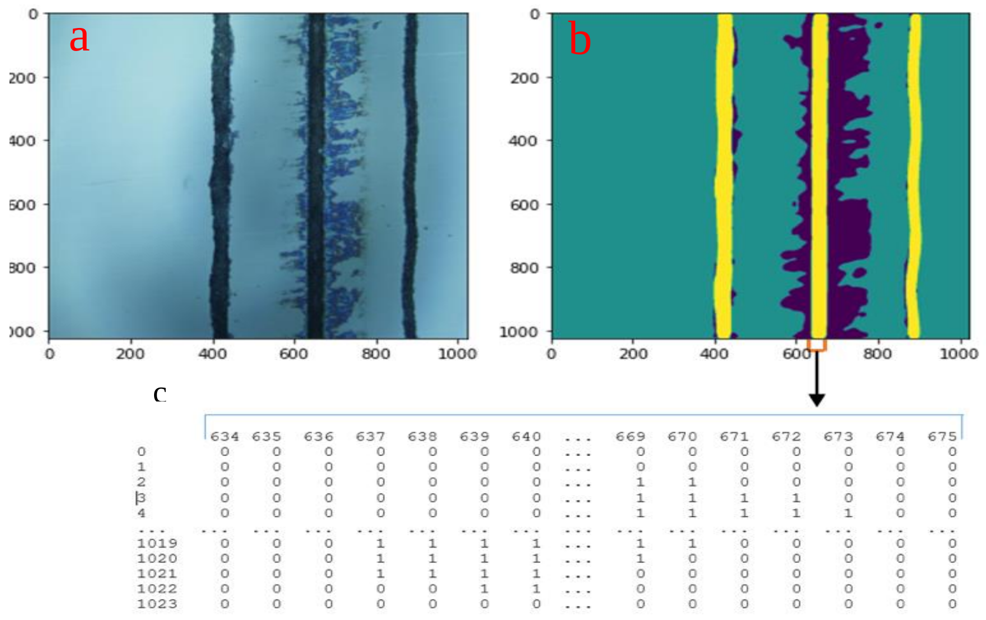
and $\bar{x}_{ij}$ is the average x_{ij} in row i. Plotting e_i gives a clear idea about the straightness of the scribe.

4.3. Results and discussion

The proposed study uses different scribing conditions causing different quality issues such as debris, fluctuation, and very thin or very thick lines. The proposed TDCNN method was able to capture and help quantify the scribed lines. In the following sections, we will discuss these findings.

4.3.1. Debris measurement

Figure 4.7 (a) is an original image from scribing 3 lines and Figure 4.7 (b) shows the classified image after the use of the proposed TDCNN model. The green region in the classified image represents the background of the part, the yellow regions are scribes, and the purple ones are debris. Figure 4.7 (c) shows the classification values around the middle line. Note that there are 1024 *1024 = 220 pixels in the image. Based on this classification, pixel counting shows that the 110893 purple pixels are debris, the 833790 greens are background, and the 103893 yellows are scribes. This means 79.5 percent of the shape is background, 9.9 percent is scribe, and 10.6 percent is debris (mostly because of the second scribe line).



```
      634 635  636  637  638  639  640   ...  669  670  671  672  673  674  675
0       0   0    0    0    0    0    0   ...    0    0    0    0    0    0    0
1       0   0    0    0    0    0    0   ...    0    0    0    0    0    0    0
2       0   0    0    0    0    0    0   ...    1    1    0    0    0    0    0
|3      0   0    0    0    0    0    0   ...    1    1    1    1    0    0    0
4       0   0    0    0    0    0    0   ...    1    1    1    1    1    0    0
...   ... ...  ...  ...  ...  ...  ...   ...  ...  ...  ...  ...  ...  ...  ...
1019    0   0    0    1    1    1    1   ...    1    1    0    0    0    0    0
1020    0   0    0    1    1    1    1   ...    1    0    0    0    0    0    0
1021    0   0    0    1    1    1    1   ...    0    0    0    0    0    0    0
1022    0   0    0    0    0    1    1   ...    0    0    0    0    0    0    0
1023    0   0    0    0    0    0    0   ...    0    0    0    0    0    0    0
```

Figure 4.7. (a) original scribed sample on top left; (b) classification using TDCNN on top right. (c) scribe classification values

4.3.2. Straightness

Figure 4.8 show straightness measurement plots for a normal (straight) line and a fluctuating line, respectively. Those plots were obtained after classifying the line using TDCNN and then quantifying straightness using equations (5-10). On top of each figure the original scribe can be seen and ei is plotted for the first tolerance zone of each scribe.

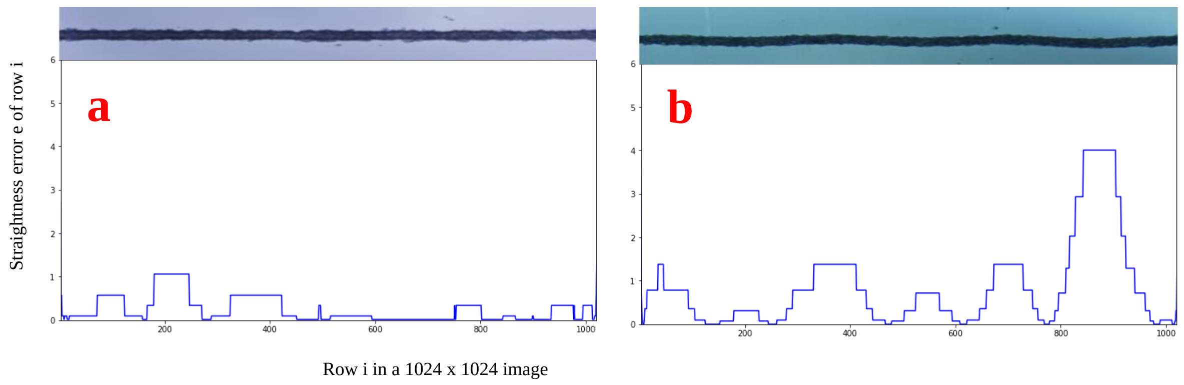


Figure 4.8. Error (e_i) plot of (a) normal line and (b) fluctuating line

The error value range for a straight line (Figure 4.8 (a)) was between 0 to 1. However, this range for a fluctuating line (Figure 4.8 (b)) was between 0 and 6. In addition, comparing Figure 4.8 (a) and (b) shows that a straight line has a smoother e_i plot with lower variations.

4.3.3. Transferability to a new case

Most DL models are designed and trained based on the assumption that experimental and environmental conditions are consistent. However, these assumptions might not hold in real life. Changes in scribing parameters and imaging conditions potentially make significant differences in the final picture. To solve this problem, the proposed model needs to be retrained in order to obtain knowledge from a new environment. In a model without transfer learning, we might need to use a big data set that includes new conditions to get appropriate accuracy. However, the proposed model can accomplish this task with only a handful of images.

In our research, to test this hypothesis, we provided a new dataset of images from a different laser and imaging conditions. This new dataset is collected from a group of lines scribed by a nanosecond laser with a wavelength of 1064 nm and a repetition rate of 10 Hz. Eight lines with different conditions were scribed. Table 4.5 shows the technical details about the second group of scribes.

Table 4.5. Scribing conditions of the second group of lines

Line number	*Pulse energy (mJ)*	*Spot size(mm)*	*Scanning speed (mm/s)*
1	202	0.7	1.5
2	202	0.7	3.0
3	202	1.2	1.5
4	202	1.2	3.0
5	415	1.2	1.5
6	415	1.2	3.0
7	415	0.7	1.5
8	415	0.7	3.0

To examine the trained TDCNN, we tried to classify the new images using the current trained model. Figure 4.9 shows the classification result on new samples without any change or retraining. Figure 4.9(a) is the original scribe image while Figure 4.9 (b) is the classification result. Note that the yellow part represents the scribe. Interestingly, the model was relatively successful in finding the background. However, we have a high rate of misclassification for the scribe. This misclassification presents a challenge for achieving our goal of automatic characterization of debris, scribe width, and scribe straightness.

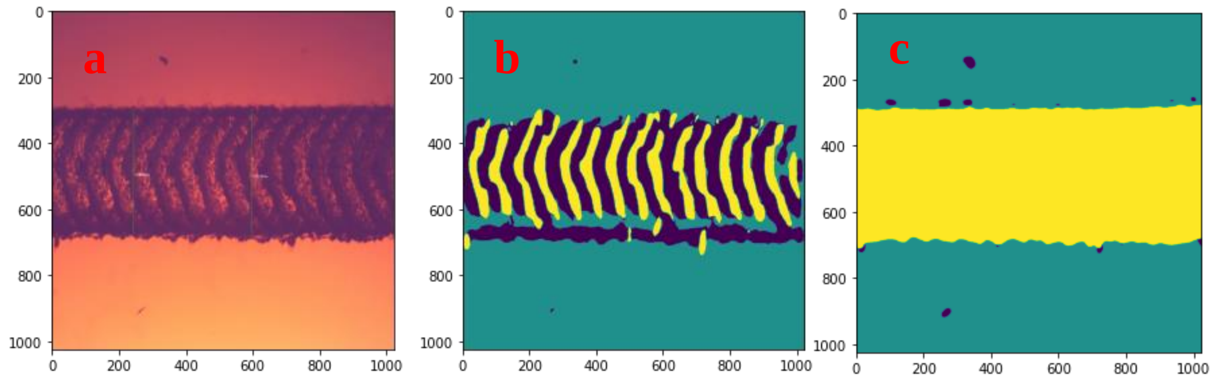


Figure 4.9. Image classification on new dataset (a) original image with different imaging setup, (b) classification result without retraining (c) classification result with

To solve this problem, we retrained the same TDCNN but used additional 6 images to create a new dataset to train a new classification model. This way the model will learn extra knowledge in addition to the knowledge it already has. Network architecture and all the training hyper parameters (e.g. learning rate) were kept the same as the first training. We manually labeled the middle of the scribe as scribe rather than background or debris in the pixels in Figure 4.9 (a). Training and validation accuracy in retraining is over 98 percent which is even better than that of the first training result. Figure 4.9 (c) shows the retraining performance on a new image. From this point, all the steps in Section 4.2.5 can be repeated and Equations (5-10) can be used to calculate scribe width and straightness.

4.4. Conclusions

In this study we proposed a novel deep transfer learning model to classify images from laser scribing with balanced performance for high accuracy and low overfitting. The classified images contain identified debris and scribes. Further image processing and algorithms are developed to quantify scribe width and straightness based on the classified images.

The proposed TDCNN has big advantages over a regular deep learning algorithm without transfer learning. First, while all regular deep learning models require a huge amount of image data and long training time for model accuracy, the proposed model requires only a few image samples for adapting new situations. This transfer learning feature in the proposed framework saves substantial effort and time in data preparation and labeling and leads to a much shorter time in the modeling and training phase. With the training of only 5 million parameters compared to the current well-known architecture (e.g., VGG16 with more than 15 million trainable parameters), the proposed model has less complexity in the newly developed portion. Second, the proposed TL architecture is flexible in that adding new information based on new scribing conditions can be achieved with minimum effort and still retains the same performance. Usually working with a small dataset increases the chance of overfitting. In the proposed TDCNN architecture, we used several layers of batch normalization and dropout to overcome this challenge. These layers helped reduce (if not remove) overfitting.

The main idea in re-using pre-trained layers is transferring general knowledge and patterns such as edges, corners, dots etc. from other labeled images. Thus, only most common features are needed from the transferred layers, specifically, the number of filters were reduced to 64. Facing a new and specific problem, 512 filters were added to capture larger combinations of patterns. For future research, more complicated pattern combinations may be captured and studied by adding new layer modules.

The proposed TDCNN model enables the characterization of laser scribing quality using just a small number of sample images and reaches the accuracy as high as 96 percent. The scribe images in this study had mainly three features of concern: debris, width variation, and straightness. The quality measures on these characteristics pave the way to track all these features automatically and enable the possibility of real-time process control as a logical next step. Although the proposed method is able to measure and quantify debris with any scribe shape, quantifying straightness and width is only limited to vertical lines and further study is needed for non-vertical lines. This study only focuses on laser scribing on silicon wafers. We expect the same framework can be extended to different materials such as solar photovoltaic thin film. However, different quality issues such as cracks may arise that also require future research. Highly unbalanced data was a significant limitation in this study. We will tackle this issue using the other state-of-the-art methods such as Differentially Private Generative Adversarial Networks in future studies as well. Finally, the case study in section 4.3.3 demonstrates that the trained TDCNN model can be transferred to a new case to improve its original performance. We expect the trained models may also be transferred to other laser related processes. Future research is much needed to confirm this hypothesis.

5. Conclusion

5.1. Summary

In this dissertation, different ML models have been studied for image segmentation applied to a wide range of applications from manufacturing process monitoring to soil cover classification. The advent of machine vision coupled with ML has enabled the monitoring and characterization of manufacturing processes as well as natural resource management. ML models driven by data have won massive popularity in recent years without physical models or expert knowledge. Although such an approach using black box ML for image-based process monitoring has brought considerable benefits, such as lower costs, and production efficiency in monitoring, it has faced significant challenges such as large data requirements, inability to produce physically consistent results, and the lack of generalizability for out-of-sample scenarios. Collecting image data may not be a big challenge anymore but pre-processing and labeling of training images is persistently a major issue. This data preparation stage is the most time-consuming and costly step in any machine vision/DL applications.

To show effectiveness of image-based process monitoring and how it can be utilized in statistical process monitoring, in chapter two we proposed a GBC to classify layer-wise images in 3D printing parts in a condition where unsupervised ML models failed to segment images due to lightning and low accuracy issue. The proposed framework first starts with a self-start control chart images that is collected using an overhead camera after a printer finishes each layer in the beginning and then switch to a cluster-charting approach after enough good parts are printed. The proposed method alleviates the need of the traditional control charting phase I requirement where at least 20 to 25 parts were recommended to establish control limits.

The classification problem of a 3D printed part versus background is relatively a simple classification problem even though unsupervised algorithms failed to provide a satisfactory result. Real world classification problems might be even more complicated where there is no specific color or shape difference between different classes that needs to be classified. Soil cover classification was the example we studied in this dissertation to classify green canopy, stubble, and soil. Despite green canopy classification which is relatively a simpler task, classifying soil and stubble is challenging due to color similarities and lack of specific geometrical shape. In chapter three, we created a benchmark dataset of semantically segmented downward-facing images obtained from agricultural fields include 3944 labeled images and then test the accuracy of a supervised DCNN to quantify the percentage of stubble, live vegetation, and bare soil using the generated dataset. The DCNN was trained for 3000 epochs and achieved 84 percent accuracy with 0.85 F1-score for soil and 0.77 F1-score for stubble.

The DCNN that was used in soil cover classification was able to achieve a satisfactory accuracy needed for agriculture and soil conservation purposes. However, frustration in the process of preprocessing and labeling those 3944 images was undeniable and perhaps the most challenging and painful part is re-doing all the steps (i.e., image collection, image preprocessing and labeling, data modeling and training) from scratch for a new problem and domain. In chapter four we proposed a novel deep transfer learning model to classify images from laser scribing with balanced performance for high accuracy and low overfitting. The classified images contain identified debris and scribes. Further image processing and algorithms are developed to quantify scribe width and straightness based on the classified images. The proposed TDCNN has big advantages over a regular deep learning algorithm without transfer learning. First, while all regular deep learning models require a huge amount of image data and long training time for model accuracy, the proposed model requires only a few image samples for adapting new situations. This transfer learning feature in the proposed framework saves substantial effort and time in data preparation and labeling and leads to a much shorter time in the modeling and training phase. The proposed TDCNN model enables the characterization of laser scribing quality using just a small number of sample images and reaches the accuracy as high as 96 percent. The scribe images in this study had mainly three features of concern: debris, width variation, and straightness. The quality measures on these characteristics pave the way to track all these features automatically and enable the possibility of real-time process control as a logical next step.

5.2. Future studies

In this research, we studied a data-driven approach for image segmentation where we collect, analyze, and extract insight from data in a specific domain. The goal was to find the best model that provides best classification under different scenarios such as capturing changes in early stage, classifying complicated segmentation problems, and segmentation using a few numbers of images as possible. This approach, where finding the best model is at the center of the problem, is also known as model-centric problem. However, along with a model-centric model it is also important to focus on a "data-centric" approach where data is the primary asset. Focusing on data-centric allows us to collect different perspective of data. For example, in 3D printing, if printed parts usually appear in the center of the image, it is hard for classifier to segment with high accuracy, if the defect is recognizable from other different angles. Additionally, focusing on data-centric is essential in improving trained model with diversity of data.

The TL approach studied in this dissertation has showed a promising improvement for reductions of training samples. However, a successful implementation of TL requires a strong similar pre-trained network to transfer the knowledge. In this research, we transferred knowledge from a pre-trained VGG16 that was trained over the ImageNet dataset. The only similarity between our domain (i.e., laser scribing) and ImageNet is that both domains are images. For future study, we recommend a multiple stages transfer learning method where the general knowledge will be transferred from a general domain such as ImageNet to a similar domain to create a good similar pre-trained network. Then having a high quality pre-trained network, we can transfer the knowledge to the target domain faster by further reduction of the training images in the problem domain of interest.

In manufacturing, our goal is improvement of manufacturing processes through process monitoring and quality control. We propose three steps of creating a pre-trained model, domain adaptation, and physics-based ML, and parameters optimization. For creating a pre-trained model, the key point in feature adaption and transfer learning is having a good pre-trained network. For future study, we propose to use generative adversarial networks (GANs) and physics-based simulation to create a pre-trained model that includes a pre-trained network using data without any experimental results. By design, the generated data and pre-trained network are not supposed to be very accurate but could create a similar network using thousands of data points to accelerate training process. GANs are type of ML methods that contains two separate neural networks where the first network is called generator and it receives a random input vector. Discriminator is the second neural network in GANs that serves as a loss function for the generator. GANs were shown a high capability to accurately capture solution manifolds of partial differential equations (PDEs) parametrized by physical parameters [173], and generating spatio-temporal super-resolution [174].

The proposed integration of the domain adaptation and physics-based ML is summarized in Figure 5.1. Given the drivers which is our generated data using scientific formula and GAN model, we will create pre-trained model f_{phy} to predict Y_{phy}. The pre-trained model f_{phy} is a function based on possible understood parameters (i.e., trained parameters). The pre-trained model f_{phy} will be again trained using experimental data and measurements using a wider range of parameters to predict output quality Y_{pred}. Finally, based on the physics-based ML prediction, we recommend running an appropriate parameter optimization model based on the experimental design such as steepest descend gradient method, Bayesian optimization, and reinforcement leaning to find best parameters as recommendation for next experiment. After finding recommended parameters, the three steps should repeat until finding best parameter settings.

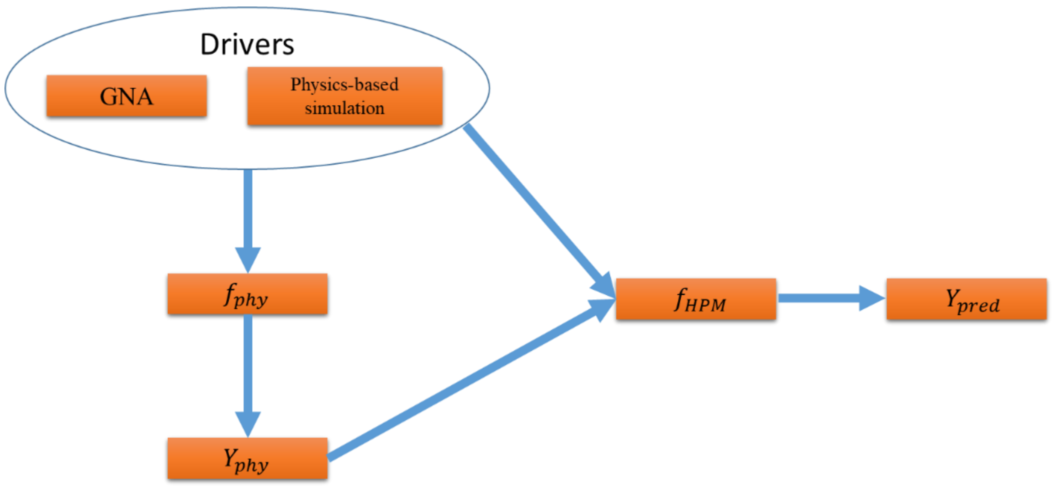


Figure 5.1. Hybrid Physics-Based Domain Adaption Model

For example, process monitoring of 3D freeze inkjet printing of aerogel could be done so that the droplet frequency can be optimized as an in-control parameter. This challenge is identification of the dimensions and shape of the printed aerogel lines. A two-stage TL model could be applied to classify printed lines from the background. In the first stage, basic image classification knowledge from the pre-trained VGG16 [127] will be transferred to a labeled laser scribing image data. The combined deep learning model is then could be used to train the model (i.e. additional layers) for the 3D inkjet aerogel printing problem. Finally, the classified images will be used to measure the width and height of the printed lines to determine the optimal printing frequency.

References

[1] A. L. Bowler and N. J. Watson, “Transfer learning for process monitoring using reflection-mode ultrasonic sensing,” *Ultrasonics*, vol. 115, p. 106468, Aug. 2021, doi: 10.1016/j.ultras.2021.106468.

[2] S. Borra, R. Thanki, and N. Dey, *Satellite image analysis : clustering and classification, studies in computational intelligence*. Singapore: Springer Singapore, 2019. doi: 10.1007/978-981-13-6424-2.

[3] H. Yu and J. F. MacGregor, “Multivariate image analysis and regression for prediction of coating content and distribution in the production of snack foods,” *Chemometrics and Intelligent Laboratory Systems*, vol. 67, no. 2, pp. 125–144, Aug. 2003, doi: 10.1016/S0169-7439(03)00065-0.

[4] H. T. Do, V. Raghavan, L. X. Truong, and G. Yonezawa, “Multi-scale object-based fuzzy classification for LULC mapping from optical satellite images,” *Spatial Information Research*, vol. 27, no. 2, pp. 247–257, 2019, doi: 10.1007/s41324-019-00240-w.

[5] E. Bellon *et al.*, “Experimental Teleradiology. Novel Telematics Services Using Image Processing, Hypermedia and Remote Cooperation to Improve Image-Based Medical Decision Making,” *Journal of Telemedicine and Telecare*, vol. 1, no. 2, pp. 100–110, Jun. 1995, doi: 10.1177/1357633X9500100206.

[6] C. Studholme, D. L. G. Hill, and D. J. Hawkes, “An overlap invariant entropy measure of 3D medical image alignment,” *Pattern Recognition*, vol. 32, no. 1, pp. 71–86, Jan. 1999, doi: 10.1016/S0031-3203(98)00091-0.

[7] D. Balageas, C.-P. Fritzen, and A. Gemes, Eds., *Structural Health Monitoring*. London, UK: ISTE, 2006. doi: 10.1002/9780470612071.

[8] H. Yan, K. Paynabar, and J. Shi, “Image-based process monitoring using low-rank tensor decomposition,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 1, pp. 216–227, 2015, doi: 10.1109/TASE.2014.2327029.

[9] G. Szatvanyi, C. Duchesne, and G. Bartolacci, “Multivariate Image Analysis of Flames for Product Quality and Combustion Control in Rotary Kilns,” *Industrial & Engineering Chemistry Research*, vol. 45, no. 13, pp. 4706–4715, Jun. 2006, doi: 10.1021/ie051336q.

[10] Y. Lyu, J. Chen, and Z. Song, “Image-based process monitoring using deep learning framework,” *Chemometrics and Intelligent Laboratory Systems*, vol. 189, no. January 2018, pp. 8–17, 2019, doi: 10.1016/j.chemolab.2019.03.008.

[11] F. M. Megahed, W. H. Woodall, and J. A. Camelio, “A review and perspective on control charting with image data,” *Journal of Quality Technology*, vol. 43, no. 2, pp. 83–98, 2011, doi: 10.1080/00224065.2011.11917848.

[12] H. Yan, K. Paynabar, and J. Shi, “Image-Based Process Monitoring Using Low-Rank Tensor Decomposition,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 1, pp. 216–227, Jan. 2015, doi: 10.1109/TASE.2014.2327029.

[13] Z. He, L. Zuo, M. Zhang, and F. M. Megahed, “An image-based multivariate generalized likelihood ratio control chart for detecting and diagnosing multiple faults in manufactured products,” *International Journal of Production Research*, vol. 54, no. 6, pp. 1771–1784, Mar. 2016, doi: 10.1080/00207543.2015.1062569.

[14] H.-C. L. Bernard C. Jiang, Chien-Chih Wang, “Liquid crystal display surface uniformity defect inspection using analysis of variance and exponentially weighted moving average techniques,” *International Journal of Production Research*, vol. 43, no. 1, pp. 67–80, Jan. 2005, doi: 10.1080/00207540412331285832.

[15] B. C. Jiang and S. J. Jiang, “Machine vision based inspection of oil seals,” *Journal of Manufacturing Systems*, vol. 17, no. 3, pp. 159–166, Jan. 1998, doi: 10.1016/S0278-6125(98)80058-7.

[16] C. J. Lu and D. M. Tsai, “Automatic defect inspection for LCDs using singular value decomposition,” *International Journal of Advanced Manufacturing Technology*, vol. 25, no. 1–2, pp. 53–61, 2005, doi: 10.1007/s00170-003-1832-6.

[17] H.-D. Lin and S. W. Chiu, “Computer-Aided Vision System for MURA-Type Defect Inspection in Liquid Crystal Displays,” 2006, pp. 442–452. doi: 10.1007/11949534_44.

[18] M. Tunák and A. Linka, “Directional Defects in Fabrics,” *Research Journal of Textile and Apparel*, vol. 12, no. 2, pp. 13–22, May 2008, doi: 10.1108/RJTA-12-02-2008-B002.

[19] J. M. Armingol, J. Otamendi, A. De La Escalera, J. M. Pastor, and F. J. Rodriguez, “Statistical pattern modeling in vision-based quality control systems,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 37, no. 3, pp. 321–336, 2003, doi: 10.1023/A:1025489610281.

[20] K. Wang and F. Tsung, “Using Profile Monitoring Techniques for a Data-rich Environment with Huge Sample Size,” *Quality and Reliability Engineering International*, vol. 21, no. 7, pp. 677–688, Nov. 2005, doi: 10.1002/qre.711.

[21] F. M. Megahed, L. J. Wells, J. A. Camelio, and W. H. Woodall, “A Spatiotemporal Method for the Monitoring of Image Data,” *Quality and Reliability Engineering International*, vol. 28, no. 8, pp. 967–980, Dec. 2012, doi: 10.1002/qre.1287.

[22] S. Bozinovski, “Reminder of the First Paper on Transfer Learning in Neural Networks, 1976,” *Informatica*, vol. 44, no. 3, Sep. 2020, doi: 10.31449/inf.v44i3.2828.

[23] J. N. Kani and A. H. Elsheikh, “DR-RNN: A deep residual recurrent neural network for model reduction,” Sep. 2017.

[24] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet : A deep convolutional encoder-decoder architecture for image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017, doi: 10.1109/TPAMI.2016.2644615.

[25] T. J. Sejnowski, *Neural Information Processing Systems*. 2019. doi: 10.7551/mitpress/11474.003.0014.

[26] E. Bauer, “An Empirical Comparison of Voting Classification Algorithms : Bagging , Boosting , and Variants,” *Science (1979)*, vol. 38, no. 1998, pp. 1–38, 2011, doi: https://doi.org/10.1023/A:1007515423169.

[27] M. Bosch, K. Foster, G. Christie, S. Wang, G. D. Hager, and M. Brown, “Semantic Stereo for Incidental Satellite Images,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2019, pp. 1524–1532. doi: 10.1109/WACV.2019.00167.

[28] L. Li, Y. Wu, and M. Ye, “Multi-class Image Classification Based on Fast Stochastic Gradient Boosting Fast stochastic gradient boosting algorithm,” vol. 38, pp. 145–153, 2014.

[29] X. Li, L. Zhang, B. Du, L. Zhang, and Q. Shi, “Iterative Reweighting Heterogeneous Transfer Learning Framework for Supervised Remote Sensing Image Classification,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 5, pp. 2022–2035, 2017, doi: 10.1109/JSTARS.2016.2646138.

[30] Harsh Sikka, “Transfer Learning Will Radically Change Machine Learning for Engineers,” 2018. https://medium.com/modeldepot/transfer-learning-will-radically-change-machine-learning-for-engineers-78732b2bb415

[31] S. Sarangi, M. Sahidullah, and G. Saha, “Optimization of data-driven filterbank for automatic speaker verification,” *Digital Signal Processing*, vol. 104, p. 102795, Sep. 2020, doi: 10.1016/j.dsp.2020.102795.

[32] S. J. Pan and Q. Yang, “A Survey on Transfer Learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[33] D. Sarkar, R. Bali, and T. Ghosh, *Hands On Transfer Learning with Python Implement Advanced Deep Learning and Neural Network Models Using TensorFlow and Keras*. Packt Publishing Ltd., 2018.

[34] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A Survey on Deep Transfer Learning,” 2018, pp. 270–279. doi: 10.1007/978-3-030-01424-7_27.

[35] S. Niu, Y. Liu, J. Wang, H. Song, and S. Member, “A Decade Survey of Transfer Learning (2010 – 2020),” vol. 1, no. 2, pp. 151–166, 2021.

[36] Y. Yao and G. Doretto, “Boosting for transfer learning with multiple sources,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010, pp. 1855–1862. doi: 10.1109/CVPR.2010.5539857.

[37] R. Xia, J. Yu, F. Xu, and S. Wang, “Instance-Based Domain Adaptation in NLP via In-Target-Domain Logistic Approximation,” in *AAAI Conference on Artificial Intelligence*, 2014, pp. 1600–1606.

[38] R. Xia, X. Hu, J. Lu, J. Yang, and C. Zong, “Instance selection and instance weighting for cross-domain sentiment classification via PU learning,” 2013.

[39] F. Xu, J. Yu, and R. Xia, “Instance-based Domain Adaptation via Multiclustering Logistic Approximation,” *IEEE Intelligent Systems*, vol. 33, no. 1, pp. 78–88, Jan. 2018, doi: 10.1109/MIS.2018.012001555.

[40] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, “Domain Adaptation via Transfer Component Analysis,” *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, Feb. 2011, doi: 10.1109/TNN.2010.2091281.

[41] H. Zhu, M. Long, J. Wang, and Y. Cao, “Deep Hashing Network for Efficient Similarity Retrieval ∗,” in *Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2415–2421.

[42] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” Nov. 2014.

[43] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014, pp. 1717–1724. doi: 10.1109/CVPR.2014.222.

[44] M. Najjartabar Bisheh, G. R. Nasiri, E. Esmaeili, H. Davoudpour, and S. I. Chang, “A new supply chain distribution network design for two classes of customers using transfer recurrent neural network,” *International Journal of System Assurance Engineering and Management*, May 2022, doi: 10.1007/s13198-022-01670-w.

[45] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” Jun. 2014.

[46] R. Mutegeki and D. S. Han, “Feature-Representation Transfer Learning for Human Activity Recognition,” in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct. 2019, pp. 18–20. doi: 10.1109/ICTC46691.2019.8939979.

[47] S. Ghassemi, A. Fiandrotti, G. Francini, and E. Magli, “Learning and adapting robust features for satellite image segmentation on heterogeneous data sets,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 6517–6529, 2019, doi: 10.1109/TGRS.2019.2906689.

[48] M. Ferguson, R. Ak, Y. T. T. Lee, and K. H. Law, “Detection and segmentation of manufacturing defects with convolutional neural networks and transfer learning,” *Smart and Sustainable Manufacturing Systems*, vol. 2, no. 1, pp. 137–164, 2018, doi: 10.1520/SSMS20180033.

[49] J. Guo, J. Wu, Z. Sun, J. Long, and S. Zhang, “Fault Diagnosis of Delta 3D Printers Using Transfer Support Vector Machine with Attitude Signals,” *IEEE Access*, vol. 7, pp. 40359–40368, 2019, doi: 10.1109/ACCESS.2019.2905264.

[50] T. Han, C. Liu, W. Yang, and D. Jiang, “Deep transfer network with joint distribution adaptation: A new intelligent fault diagnosis framework for industry application,” *ISA Transactions*, vol. 97, no. xxxx, pp. 269–281, 2020, doi: 10.1016/j.isatra.2019.08.012.

[51] W. Jiao, Q. Wang, Y. Cheng, and Y. M. Zhang, “End-to-end prediction of weld penetration: A deep learning and transfer learning based method,” *Journal of Manufacturing Processes*, vol. 63, no. January 2020, pp. 191–197, 2021, doi: 10.1016/j.jmapro.2020.01.044.

[52] Y. Liao, I. Ragai, Z. Huang, and S. Kerner, “Manufacturing process monitoring using time-frequency representation and transfer learning of deep neural networks,” *Journal of Manufacturing Processes*, vol. 68, no. PA, pp. 231–248, 2021, doi: 10.1016/j.jmapro.2021.05.046.

[53] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A Survey on Deep Transfer Learning,” 2018, pp. 270–279. doi: 10.1007/978-3-030-01424-7_27.

[54] K. He and J. Sun, “Convolutional neural networks at constrained time cost,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 5353–5360. doi: 10.1109/CVPR.2015.7299173.

[55] A. Perez, S. Ganguli, S. Ermon, G. Azzari, M. Burke, and D. Lobell, “Semi-Supervised Multitask Learning on Multispectral Satellite Images Using Wasserstein Generative Adversarial Networks (GANs) for Predicting Poverty,” 2019.

[56] C. Lei, “Deep Reinforcement Learning,” 2021, pp. 217–243. doi: 10.1007/978-981-16-2233-5_10.

[57] Y. Li, B. We, D. Q-network, and D. R. Learning, “D r l : a o,” pp. 1–85.

[58] J. Brownlee, *Better deep learning: train faster, reduce overfitting, and make better predictions*. Machine Learning Mastery, 2018.

[59] gene Brown, “Difference Between Deep Learning and Reinforcement Learning,” 2019. http://www.differencebetween.net/technology/difference-between-deep-learning-and-reinforcement-learning/

[60] M. Sewak, *Deep reinforcement learning*. Springer, 2019.

[61] F. Chollet, “Deep learning with Python. Shelter Island Manning.” New York: Manning Publications Company, 2017.

[62] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[63] M. Najjartabar Bisheh, S. I. Chang, and S. Lei, “A layer-by-layer quality monitoring framework for 3D printing,” *Computers and Industrial Engineering*, vol. 157, no. March, p. 107314, 2021, doi: 10.1016/j.cie.2021.107314.

[64] Douglas C. Montgomery, *Introduction to Statistical Quality Control*, Seventh. John Wiley & Sons, Inc., 2013.

[65] D. Nahitiya, M. N. Bisheh, R. P. Lollato, and A. Patrignani, “Preliminary Classification of Soil, Plant, and Residue Cover Using Convolutional Neural Networks,” *Kansas*

Agricultural Experiment Station Research Reports, vol. 7, no. 5, Jan. 2021, doi: 10.4148/2378-5977.8081.

[66] M. N. Bisheh, X. Wang, S. I. Chang, S. Lei, and J. Ma, “Image-based characterization of laser scribing quality using transfer learning,” *Journal of Intelligent Manufacturing*, Mar. 2022, doi: 10.1007/s10845-022-01926-z.

[67] A. Beltagui, A. Rosli, and M. Candi, “Exaptation in a digital innovation ecosystem: The disruptive impacts of 3D printing,” *Research Policy*, vol. 49, no. 1, 2020, doi: 10.1016/j.respol.2019.103833.

[68] S. J. Hu, “Evolving Paradigms of Manufacturing: From Mass Production to Mass Customization and Personalization,” *Procedia CIRP*, vol. 7, pp. 3–8, 2013, doi: 10.1016/j.procir.2013.05.002.

[69] M. Amini, S. I. Chang, and P. Rao, “A cybermanufacturing and AI framework for laser powder bed fusion (LPBF) additive manufacturing process,” *Manufacturing Letters*, vol. 21, pp. 41–44, Aug. 2019, doi: 10.1016/j.mfglet.2019.08.007.

[70] M. Amini and S. Chang, “Process Monitoring of 3D Metal Printing in Industrial Scale,” Jun. 2018. doi: 10.1115/MSEC2018-6332.

[71] M. Amini and S. I. Chang, “MLCPM: A process monitoring framework for 3D metal printing in industrial scale,” *Computers & Industrial Engineering*, vol. 124, pp. 322–330, Oct. 2018, doi: 10.1016/j.cie.2018.07.041.

[72] N. Shahrubudin, T. C. Lee, and R. Ramlan, “An overview on 3D printing technology: Technological, materials, and applications,” *Procedia Manufacturing*, vol. 35, pp. 1286–1296, 2019, doi: 10.1016/j.promfg.2019.06.089.

[73] V. Sreehitha, “Impact of 3D Printing in Automotive Industries,” *International Journal of Mechanical And Production Engineering*, no. 5, pp. 2320–2092, 2017.

[74] R. Manghnani, “An Exploratory Study: The impact of Additive Manufacturing on the Automobile Industry,” *International Journal of Current Engineering and Technology*, vol. 5, no. 5, pp. 3407–3410, 2015.

[75] H. S. Lim, “Development of 3D Printed Shoe Designs Using Traditional Muntin Patterns,” *Fashion & Textile Research Journal*, vol. 19, no. 2, pp. 134–139, Apr. 2017, doi: 10.5805/SFTI.2017.19.2.134.

[76] F. T. Piller, E. Lindgens, and F. Steiner, “Mass Customization at Adidas: Three Strategic Capabilities to Implement Mass Customization,” *SSRN Electronic Journal,* pp. 1–22, 2012, doi: 10.2139/ssrn.1994981.

[77] ISO/ASTM, “ISO/ASTM 52900: Additive manufacturing - General principles - Terminology,” *International Standard*, vol. 5, pp. 1–26, 2015.

[78] C. Weller, R. Kleer, and F. T. Piller, “Economic implications of 3D printing: Market structure models in light of additive manufacturing revisited,” *International Journal of Production Economics*, vol. 164, pp. 43–56, 2015, doi: 10.1016/j.ijpe.2015.02.020.

[79] F. Imani, B. Yao, R. Chen, P. Rao, and H. Yang, “Joint Multifractal and Lacunarity Analysis of Image Profiles for Manufacturing Quality Control,” *Journal of Manufacturing Science and Engineering*, vol. 141, no. 4, Apr. 2019, doi: 10.1115/1.4042579.

[80] L. M. Q. Abualigah, *Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering*, vol. 816. Cham: Springer International Publishing, 2019. doi: 10.1007/978-3-030-10674-4.

[81] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, “A new feature selection method to improve the document clustering using particle swarm optimization algorithm,” *Journal of Computational Science,* vol. 25, pp. 456–466, Mar. 2018, doi: 10.1016/j.jocs.2017.07.018.

[82] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, “Hybrid clustering analysis using improved krill herd algorithm,” *Applied Intelligence*, vol. 48, no. 11, pp. 4047–4071, Nov. 2018, doi: 10.1007/s10489-018-1190-6.

[83] SASAM, “Support Action for Standardisation in Additive Manufacturing (SASAM) (FP7–NMP–2012-CSA-6),” p. 26, 2014.

[84] E. N. Malamas, E. G. M. Petrakis, M. Zervakis, L. Petit, and J.-D. Legat, “A survey on industrial vision systems, applications and tools,” *Image and Vision Computing*, vol. 21, no. 2, pp. 171–188, Feb. 2003, doi: 10.1016/S0262-8856(02)00152-X.

[85] X. Guo, Z. He, and H. Chen, “A Real-Time Contrasts Method for Monitoring Image Data,” in *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA),* Apr. 2019, pp. 354–361. doi: 10.1109/IEA.2019.8715222.

[86] R. L. R. L. Horst and M. Negin, “Vision system for high-resolution dimensional measurements and on-line SPC: web process application,” *IEEE Transactions on Industry Applications*, vol. 28, no. 4, pp. 993–997, 1992, doi: 10.1109/28.148468.

[87] M. Koosha, R. Noorossana, and F. Megahed, “Statistical process monitoring via image data using wavelets,” *Quality and Reliability Engineering International*, vol. 33, no. 8, pp. 2059–2073, Dec. 2017, doi: 10.1002/qre.2167.

[88] U. Delli and S. Chang, “Automated Process Monitoring in 3D Printing Using Supervised Machine Learning,” *Procedia Manufacturing*, vol. 26, pp. 865–870, 2018, doi: 10.1016/j.promfg.2018.07.111.

[89] F. Imani, A. Gaikwad, M. Montazeri, P. Rao, H. Yang, and E. Reutzel, “Process Mapping and In-Process Monitoring of Porosity in Laser Powder Bed Fusion Using Layerwise Optical Imaging,” *Journal of Manufacturing Science and Engineering*, vol. 140, no. 10, Oct. 2018, doi: 10.1115/1.4040615.

[90] F. Imani, R. Chen, E. Diewald, E. Reutzel, and H. Yang, “Deep Learning of Variant Geometry in Layerwise Imaging Profiles for Additive Manufacturing Quality Control,” *Journal of Manufacturing Science and Engineering*, vol. 141, no. 11, Nov. 2019, doi: 10.1115/1.4044420.

[91] L. Zuo, Z. He, and M. Zhang, “An EWMA and region growing based control chart for monitoring image data,” *Quality Technology & Quantitative Management*, vol. 17, no. 4, pp. 470–485, Jul. 2020, doi: 10.1080/16843703.2019.1682751.

[92] J. H. Sullivan, “Detection of multiple change points from clustering individual observations,” *Journal of Quality Technology*, vol. 34, no. 4, pp. 371–383, 2002, doi: 10.1080/00224065.2002.11980170.

[93] A. Patrignani and T. E. Ochsner, “Canopeo: A powerful new tool for measuring fractional green canopy cover,” *Agronomy Journal*, vol. 107, no. 6, pp. 2312–2320, 2015, doi: 10.2134/agronj15.0150.

[94] D. M. Hawkins, “Self-Starting Cusum Charts for Location and Scale,” *The Statistician*, vol. 36, no. 4, p. 299, 1987, doi: 10.2307/2348827.

[95] S. A. Daryabari, B. Malmir, and A. Amiri, “Monitoring Bernoulli processes considering measurement errors and learning effect,” *Quality and Reliability Engineering International*, vol. 35, no. 4, pp. 1129–1143, Jun. 2019, doi: 10.1002/qre.2449.

[96] S. I. Chang *et al.*, “Retrospective analysis for phase I statistical process control and process capability study using revised sample entropy,” *Neural Computing and Applications*, vol. 31, no. 11, pp. 7415–7428, Nov. 2019, doi: 10.1007/s00521-018-3556-4.

[97] S. I. Chang, B. Tavakkol, S.-H. Chou, and T.-R. Tsai, “Real-time detection of wave profile changes,” *Computers & Industrial Engineering*, vol. 75, pp. 187–199, Sep. 2014, doi: 10.1016/j.cie.2014.05.020.

[98] Z. Yang, H. Minggagud, T. Baoyin, and F. Y. Li, “Plant production decreases whereas nutrients concentration increases in response to the decrease of mowing stubble height,” *Journal of Environmental Management*, vol. 253, p. 109745, Jan. 2020, doi: 10.1016/j.jenvman.2019.109745.

[99] A. Bot and J. Benites, *The importance of soil organic matter Key to drought-resistant soil and sustained food production*. 2005.

[100] A. L. Black and D. L. Tanaka, “A conservation tillage-cropping systems study in the Northern Great Plains of the United States,” in *Soil Organic Matter in Temperate Agroecosystems*, 2019, pp. 335–342. doi: https://doi.org/10.1201/9780367811693.

[101] C. R. Fenster, “Stubble mulching with various types of machinery,” *Soil Science Society of America Journal*, vol. 24, no. 6, pp. 518–523, Nov. 1960, doi: 10.2136/sssaj1960.03615995002400060030x.

[102] M.-S. Turmel, A. Speratti, F. Baudron, N. Verhulst, and B. Govaerts, “Crop residue management and soil health: A systems analysis,” *Agricultural Systems*, vol. 134, pp. 6–16, Mar. 2015, doi: 10.1016/j.agsy.2014.05.009.

[103] Brun, Lynn, J. W. Enz, J. K. Larsen, and C. Fanning, “Springtime evaporation from bare and stubble-covered soil,” *Journal of Soil and Water Conservation*, vol. 41, no. 2, pp. 120–122, 1984.

[104] Y. Shen, N. McLaughlin, X. Zhang, M. Xu, and A. Liang, “Effect of tillage and crop residue on soil temperature following planting for a Black soil in Northeast China,” *Scientific Reports*, vol. 8, no. 1, p. 4500, Dec. 2018, doi: 10.1038/s41598-018-22822-8.

[105] M. Quemada and C. S. T. Daughtry, “Spectral indices to improve crop residue cover estimation under varying moisture conditions,” *Remote Sensing*, vol. 8, no. 8, 2016, doi: 10.3390/rs8080660.

[106] R. Lal, “Is crop residue a waste,” *Journal of Soil and Water Conservation*, vol. 59, no. 6, pp. 136A-139A, 2004.

[107] H. Blanco-Canqui and R. Lal, “Crop Residue Removal Impacts on Soil Productivity and Environmental Quality,” *Critical Reviews in Plant Sciences*, vol. 28, no. 3, pp. 139–163, Apr. 2009, doi: 10.1080/07352680902776507.

[108] B. K. Richards, M. F. Walter, and R. E. Muck, “Variation in line of crop residue cover,” *Journal of Soil and Water Conservation*, vol. 39, no. 1, pp. 60–61, 1982.

[109] D. P. Shelton, P. J. Jasa, D. P. Shelton, and E. A. Engineer, “Estimating Percent Residue Cover Using the Line-Transect Method,” *Biological Systems Engineering: Papers and Publications*, 2009.

[110] D. P. Shelton *et al.*, “NebGuide Using the Line-Transect Method to Estitnate Percent Residue Cover,” *Biological Systems Engineering: Papers and Publications*, vol. 289, pp. 6–8, 1990.

[111] J. M. Laflen, M. Amemiya, and E. A. Hintz., “Measuring crop residue cover,” *Journal of Soil and Water Conservation*, vol. 36, no. 6, pp. 341–343, 1981.

[112] J. M. Molloy and C. J. Moran, “Compiling a field manual from overhead photographs for estimating crop residue cover,” *Soil Use and Management*, vol. 7, no. 4, pp. 177–183, 1991.

[113] S. J. Corak, T. C. Kaspar, and D. W. Meek, “Evaluating methods for measuring residue cover,” *J Soil Water Conserv*, pp. 70–74, 1993.

[114] A. Patrignani and T. E. Ochsner, “Canopeo: A Powerful New Tool for Measuring Fractional Green Canopy Cover,” *Agronomy Journal*, vol. 107, no. 6, pp. 2312–2320, Nov. 2015, doi: 10.2134/agronj15.0150.

[115] H. M. Easlon and A. J. Bloom, “Easy Leaf Area: Automated digital image analysis for rapid and accurate measurement of leaf area,” *Applications in Plant Sciences*, vol. 2, no. 7, p. 1400033, Jul. 2014, doi: 10.3732/apps.1400033.

[116] J. M. Paruelo, W. K. Lauenroth, and P. A. Roset, “Estimating Aboveground Plant Biomass Using a Photographic Technique,” *Journal of Range Management*, vol. 53, no. 2, p. 190, Mar. 2000, doi: 10.2307/4003281.

[117] F. J. Adamsen *et al.*, “Measuring Wheat Senescence with a Digital Camera,” *Crop Science*, vol. 39, no. 3, pp. 719–724, May 1999, doi: 10.2135/cropsci1999.0011183X003900030019x.

[118] P. Riegler-Nurscher, J. Prankl, T. Bauer, P. Strauss, and H. Prankl, “A machine learning approach for pixel wise classification of residue and vegetation cover under field

conditions," *Biosystems Engineering*, vol. 169, pp. 188–198, May 2018, doi: 10.1016/j.biosystemseng.2018.02.011.

[119] D. Ruppert, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, vol. 99, no. 466. Springer Series in Statistics, 2004. doi: 10.1198/jasa.2004.s339.

[120] A. B. Gavade and V. S. Rajpurohit, "Systematic analysis of satellite image-based land cover classification techniques: literature review and challenges," *International Journal of Computers and Applications*, vol. 0, no. 0, pp. 1–10, 2019, doi: 10.1080/1206212x.2019.1573946.

[121] A. Wang, W. Zhang, and X. Wei, "A review on weed detection using ground-based machine vision and image processing techniques," *Computers and Electronics in Agriculture*, vol. 158, pp. 226–240, 2019, doi: 10.1016/j.compag.2019.02.005.

[122] J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun, "A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network," *Computers and Electronics in Agriculture*, vol. 154, pp. 18–24, Nov. 2018, doi: 10.1016/j.compag.2018.08.048.

[123] A. Fuentes, S. Yoon, S. Kim, and D. Park, "A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition," *Sensors*, vol. 17, no. 9, p. 2022, Sep. 2017, doi: 10.3390/s17092022.

[124] A. G. Smith, J. Petersen, R. Selvan, and C. R. Rasmussen, "Segmentation of roots in soil with U-Net," *Plant Methods*, vol. 16, no. 1, p. 13, Dec. 2020, doi: 10.1186/s13007-020-0563-0.

[125] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, no. July 2017, pp. 70–90, 2018, doi: 10.1016/j.compag.2018.02.016.

[126] D. Bradley and G. Roth, "Adaptive Thresholding using the Integral Image," *Journal of Graphics Tools*, vol. 12, no. 2, pp. 13–21, Jan. 2007, doi: 10.1080/2151237X.2007.10129236.

[127] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–14, 2015.

[128] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, “Revisiting Batch Normalization For Practical Domain Adaptation,” Mar. 2016.

[129] L. Zhong, L. Hu, and H. Zhou, “Deep learning based multi-temporal crop classification,” *Remote Sensing of Environment*, vol. 221, pp. 430–443, Feb. 2019, doi: 10.1016/j.rse.2018.11.032.

[130] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 1520–1528, 2015, doi: 10.1109/ICCV.2015.178.

[131] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” pp. 234–241, 2015, doi: 10.1007/978-3-319-24574-4_28.

[132] N. Feng, “Study on MRI Medical Image Segmentation Technology Based on CNN-CRF Model,” *IEEE Access*, vol. 8, pp. 60505–60514, 2020, doi: 10.1109/ACCESS.2020.2982197.

[133] C. Yakopcic *et al.*, “Recurrent residual U-Net for medical image segmentation,” vol. 6, no. 1, 2021, doi: 10.1117/1.JMI.6.1.014006.

[134] P. Van Molle, M. De Strooper, T. Verbelen, B. Vankeirsbilck, P. Simoens, and B. Dhoedt, “Visualizing convolutional neural networks to improve decision support for skin lesion classification,” *arXiv*, pp. 1–12, 2018.

[135] S. Ku, B. E. Pieters, S. Haas, A. Bauer, Q. Ye, and U. Rau, “Electrical characterization of P3 isolation lines patterned with a UV laser incident from the film side on thin-film silicon solar cells,” *Solar Energy Materials and Solar Cells*, vol. 108, pp. 87–92, Jan. 2013, doi: 10.1016/j.solmat.2012.09.017.

[136] X. Zhao, Y. Cao, Q. Nian, Y. C. Shin, and G. Cheng, “Precise selective scribing of thin-film solar cells by a picosecond laser,” *Applied Physics A*, vol. 116, no. 2, pp. 671–681, Aug. 2014, doi: 10.1007/s00339-014-8330-6.

[137] X. Wang *et al.*, “Characterization and control of laser induced modification inside silicon,” *Journal of Laser Applications*, vol. 31, no. 2, p. 022601, May 2019, doi: 10.2351/1.5096086.

[138] K.-H. Leitz, B. Redlingshöfer, Y. Reg, A. Otto, and M. Schmidt, “Metal Ablation with Short and Ultrashort Laser Pulses,” *Physics Procedia*, vol. 12, pp. 230–238, 2011, doi: 10.1016/j.phpro.2011.03.128.

[139] X. Wang *et al.*, “Nanosecond laser writing of straight and curved waveguides in silicon with shaped beams,” *Journal of Laser Applications*, vol. 32, no. 2, p. 022002, May 2020, doi: 10.2351/1.5139973.

[140] Y. L. Yao, H. Chen, and W. Zhang, “Time scale effects in laser material removal: a review,” *The International Journal of Advanced Manufacturing Technology*, vol. 26, no. 5–6, pp. 598–608, Sep. 2005, doi: 10.1007/s00170-003-2026-y.

[141] X. Wang *et al.*, “Curved waveguides in silicon written by a shaped laser beam,” *Optics Express*, vol. 29, no. 10, p. 14201, May 2021, doi: 10.1364/OE.419074.

[142] H. Roozbahani, P. Marttinen, and A. Salminen, “Real-Time Monitoring of Laser Scribing Process of CIGS Solar Panels Utilizing High-Speed Camera,” *IEEE Photonics Technology Letters*, vol. 30, no. 20, pp. 1741–1744, 2018, doi: 10.1109/LPT.2018.2867274.

[143] H. Roozbahani, A. Salminen, and M. Manninen, “Real-time monitoring of laser scribing process utilizing high-speed camera,” vol. 2310, p. 2310, 2019, doi: 10.2351/1.5118570.

[144] Z. Y. Chua, I. H. Ahn, and S. K. Moon, “Process monitoring and inspection systems in metal additive manufacturing: Status and applications,” *International Journal of Precision Engineering and Manufacturing-Green Technology*, vol. 4, no. 2, pp. 235–245, Apr. 2017, doi: 10.1007/s40684-017-0029-7.

[145] F. Imani, A. Gaikwad, M. Montazeri, P. Rao, H. Yang, and E. Reutzel, “Process mapping and in-process monitoring of porosity in laser powder bed fusion using layerwise optical imaging,” *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 140, no. 10, Oct. 2018, doi: 10.1115/1.4040615.

[146] M. Grasso, V. Laguzza, Q. Semeraro, and B. M. Colosimo, “In-Process Monitoring of Selective Laser Melting: Spatial Detection of Defects Via Image Data Analysis,” *Journal of Manufacturing Science and Engineering*, vol. 139, no. 5, May 2017, doi: 10.1115/1.4034715.

[147] F. Imani, A. Gaikwad, M. Montazeri, P. Rao, H. Yang, and E. Reutzel, “Layerwise In-Process Quality Monitoring in Laser Powder Bed Fusion,” Jun. 2018. doi: 10.1115/MSEC2018-6477.

[148] M. Grasso, A. G. Demir, B. Previtali, and B. M. Colosimo, “In situ monitoring of selective laser melting of zinc powder via infrared imaging of the process plume,” *Robotics and*

Computer-Integrated Manufacturing, vol. 49, pp. 229–239, Feb. 2018, doi: 10.1016/j.rcim.2017.07.001.

[149] B. Yuan, B. Giera, G. Guss, I. Matthews, and S. Mcmains, “Semi-Supervised Convolutional Neural Networks for In-Situ Video Monitoring of Selective Laser Melting,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2019, pp. 744–753. doi: 10.1109/WACV.2019.00084.

[150] B. Fotovvati, S. F. Wayne, G. Lewis, and E. Asadi, “A Review on Melt-Pool Characteristics in Laser Welding of Metals,” *Advances in Materials Science and Engineering*, vol. 2018, pp. 1–18, 2018, doi: 10.1155/2018/4920718.

[151] M. Najjartabar-Bisheh, S. I. Chang, and S. Lei, “A Layer-by-Layer Quality Monitoring Framework for 3D Printing,” *Computers & Industrial Engineering*, p. 107314, Apr. 2021, doi: 10.1016/j.cie.2021.107314.

[152] S. Shevchik *et al.*, “Supervised deep learning for real-time quality monitoring of laser welding with X-ray radiographic guidance,” *Scientific Reports*, vol. 10, no. 1, p. 3389, Dec. 2020, doi: 10.1038/s41598-020-60294-x.

[153] A. Mayr *et al.*, “Evaluation of Machine Learning for Quality Monitoring of Laser Welding Using the Example of the Contacting of Hairpin Windings,” in *2018 8th International Electric Drives Production Conference (EDPC)*, Dec. 2018, pp. 1–7. doi: 10.1109/EDPC.2018.8658346.

[154] B. Zhang, K.-M. Hong, and Y. C. Shin, “Deep-learning-based porosity monitoring of laser welding process,” *Manufacturing Letters*, vol. 23, pp. 62–66, Jan. 2020, doi: 10.1016/j.mfglet.2020.01.001.

[155] C. Gonzalez-Val, A. Pallas, V. Panadeiro, and A. Rodriguez, “A convolutional approach to quality monitoring for laser manufacturing,” *Journal of Intelligent Manufacturing*, vol. 31, no. 3, pp. 789–795, Mar. 2020, doi: 10.1007/s10845-019-01495-8.

[156] S. A. Shevchik *et al.*, “Laser Welding Quality Monitoring via Graph Support Vector Machine With Data Adaptive Kernel,” *IEEE Access*, vol. 7, pp. 93108–93122, 2019, doi: 10.1109/ACCESS.2019.2927661.

[157] H. Roozbahani, A. Salminen, and M. Manninen, “Real-time online monitoring of nanosecond pulsed laser scribing process utilizing spectrometer,” *Journal of Laser Applications*, vol. 29, no. 2, p. 022208, May 2017, doi: 10.2351/1.4983520.

[158] M. Grasso, V. Laguzza, Q. Semeraro, and B. M. Colosimo, “In-Process Monitoring of Selective Laser Melting: Spatial Detection of Defects Via Image Data Analysis,” *Journal of Manufacturing Science and Engineering*, vol. 139, no. 5, May 2017, doi: 10.1115/1.4034715.

[159] D. Bradley and G. Roth, “Adaptive Thresholding using the Integral Image,” *Journal of Graphics Tools*, vol. 12, no. 2, pp. 13–21, Jan. 2007, doi: 10.1080/2151237X.2007.10129236.

[160] N. Otsu, “Threshold selection method from gray-level histograms,” *IEEE Trans Syst Man Cybern*, vol. SMC-9, no. 1, pp. 62–66, 1979, doi: 10.1109/TSMC.1979.4310076.

[161] X. Xie, J. W. K. Ho, C. Murphy, G. Kaiser, B. Xu, and T. Y. Chen, “Testing and validating machine learning classifiers by metamorphic testing,” *Journal of Systems and Software*, vol. 84, no. 4, pp. 544–558, Apr. 2011, doi: 10.1016/j.jss.2010.11.920.

[162] J. Ding, X.-H. Hu, and V. Gudivada, “A Machine Learning Based Framework for Verification and Validation of Massive Scale Image Data,” *IEEE Transactions on Big Data*, vol. 7, no. 2, pp. 451–467, Jun. 2021, doi: 10.1109/TBDATA.2017.2680460.

[163] A. Unnikrishnan, V. Sowmya, and K. P. Soman, “Deep learning architectures for land cover classification using red and near-infrared satellite images,” *Multimedia Tools and Applications*, vol. 78, no. 13, pp. 18379–18394, 2019, doi: 10.1007/s11042-019-7179-2.

[164] B. Uzkent *et al.*, “Learning to interpret satellite images using wikipedia,” *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2019-Augus, pp. 3620–3626, 2019, doi: 10.24963/ijcai.2019/502.

[165] H. K. Mousavi, M. Nazari, M. Takac, and N. Motee, “Multi-Agent Image Classification via Reinforcement Learning,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 5020–5027, 2019, doi: 10.1109/IROS40897.2019.8968129.

[166] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pp. 1–16, 2019.

[167] M. Ferguson, R. Ak, Y.-T. T. Lee, and K. H. Law, “Automatic localization of casting defects with convolutional neural networks,” in *2017 IEEE International Conference on Big Data (BIGDATA)*, 2018, no. December, pp. 1726–1735. doi: 10.1109/bigdata.2017.8258115.

[168] H. Noh, S. Hong, and B. Han, “Learning Deconvolution Network for Semantic Segmentation,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 1520–1528. doi: 10.1109/ICCV.2015.178.

[169] S. Sharma, J. E. Ball, B. Tang, D. W. Carruth, M. Doude, and M. A. Islam, “Semantic segmentation with transfer learning for off-road autonomous driving,” *Sensors (Switzerland)*, vol. 19, no. 11, pp. 1–21, 2019, doi: 10.3390/s19112577.

[170] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, “Adam Optimization Algorithm for Wide and Deep Neural Network,” *Knowledge Engineering and Data Science*, vol. 2, no. 1, p. 41, Jun. 2019, doi: 10.17977/um018v2i12019p41-46.

[171] NADCA, *NADCA Product Specification Standarts for Die Casting*. 2015.

[172] “GD&T Straightness,” *Geometric Dimensioning and Tolerancing (GD&T)*, 2014. https://www.gdandtbasics.com/straightness/

[173] M. Chu, N. Thuerey, H. P. Seidel, C. Theobalt, and R. Zayer, “Learning meaningful controls for fluids,” *ACM Transactions on Graphics*, vol. 40, no. 4, 2021, doi: 10.1145/3450626.3459845.

[174] Y. Xie, E. Franz, M. Chu, and N. Thuerey, “tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow,” Jan. 2018, doi: 10.1145/3072959.3073643.