Defect Detection for Additive Manufacturing with Machine Learning and Markov Decision Process

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Rui Li

May 2022

Abstract

Additive Manufacturing (AM) is a quickly evolving manufacturing technique in recent years. One of the most essential steps is the quality control of it. This involves the defect detection of the products, which is one of the bottlenecks that affects the high quality of AM products. One promising solution to this problem is to detect the defects in-situ and make decisions on the fly. We adopted Machine Learning (ML) algorithms for defect detection and develop a Markov Decision Process (MDP) model to make decisions for AM process. Our main purpose is to save costs and time through early termination or parameter adjustment of the printing process.

In *chapter 1*, we developed a scheme based on ML models trained. Then these models are applied to detect defects in actual production. It will save training time and costs associated with many prints for each design by using synthetic 3D point clouds rather than experimental data. Besides, a new concept called "patch" to capture macro-level information about nearby points for ML training and implementation is introduced here. Numerical comparisons of prediction results on experimental data with different shapes showed that the proposed ML-based scheme outperformed the existing Z-difference method in the literature.

In *chapter 2*, we introduced a new curvature feature based on the models in *chapter 1*, i.e., Discrete Mean Curvature Measure, to capture macro-level information beyond the distances and incorporated it into the training data for ML algorithms. This new curvature feature was demonstrated to well improve the defect detection performance (the F-measure: as high as 94%).

In *chapter 3*, we proposed an MDP model serving as the feedback mechanism to form a whole closed-loop in-situ monitoring system for AM process. We defined the general framework of the MDP model for this purpose. Then we use Value Iteration method to solve a series of cases of the model. Some

numerical analysis of the solution is also implemented to conclude some interesting phenomena in the process of solving.

In the future, it can be planned to collect experimental data to validate if the method is an applicable monitoring system for real production.

Key words: Machine Learning, Additive Manufacturing, Markov Decision Process.

Table of Contents

List of Tables

Lists of Figures

Chapter 1

Geometrical Defect Detection for Additive Manufacturing with Machine Learning Models

Abstract: *In-situ* detection of defects during additive manufacturing can reduce costs and time through early termination or parameter adjustment of the process. This study proposed a scheme based on Machine Learning (ML) models to detect geometric defects of printed objects. In this scheme, the ML models are trained with synthetic 3D point clouds with defects of varying sizes and then applied to detect defects in actual production. Using synthetic 3D point clouds rather than experimental data could save a huge amount of training time and costs associated with many prints for each design. Besides distance differences of individual points between source and target point clouds, this scheme uses a new concept called "patch" to capture macro-level information about nearby points for ML training and implementation. Numerical comparisons of prediction results on experimental data with different shapes showed that the proposed ML-based scheme outperformed the existing Z-difference method in the literature. Five ML methods（*Bagging of Trees, Gradient Boosting, Random Forest, K-nearest Neighbors* and *Linear Supported Vector Machine)* were compared under various conditions, such as different point cloud densities and defect sizes. *Bagging* and *Random Forest* were found the two best ML models regarding predictability; and the right patch size was found to be at 20. The proposed ML-based scheme is applicable to real time *in-situ* defect detection during additive manufacturing with the aid of a proper 3D data acquisition system.

Key words: Additive Manufacturing; Defect Detection; Machine Learning.

1.1 Introduction

Additive Manufacturing (AM) produces three-dimensional (3D) objects layer by layer according to the designed computer-aided design (CAD) models. AM has advantages on complicated geometric designs over traditional manufacturing (Guo & Leu, 2013) but has limitations of slow production speed, high cost, and low accuracy (I. Campbell, Bourell, & Gibson, 2012). It is estimated that superficial defects caused 10% of failures in AM-manufactured parts (R. Leach, 2011). AM defects include discontinuities and flaws, such as porosity, layer shift, inclusions, delamination and lack of fusion. All these defects may lead to geometrical inaccuracy and negatively influence mechanical properties (Mandache, 2019). Many AM parts need postprocesses to satisfy surface finish requirements. Meanwhile, AM processes normally take long time, hours or even days, to finish one part. Although AM machine and material costs are decreasing in recent years, they are still higher than traditional manufacturing (Mandache, 2019). Therefore, *in-situ* defect detection during AM processes instead of post-production measurement will help to save production time and expense through early termination or real-time parameter adjustment. *In-situ* defect detection has been a technical challenge faced by the AM community for decades (Everton, Hirsch, Stavroulakis, Leach, & Clare, 2016) (Kim, Lin, & Tseng, 2018). A major barrier is the complex and dynamic characteristics of AM (Wuest, Weimer, Irgens, & Thoben, 2016), which make *in-situ* data acquisition and analysis difficult. The real-time data acquisition system for defect detection is an important part of an *in-situ* quality control system. There are many different acquisition techniques, such as vision-based, ultrasound (Seifi, Salem, Beuth, Harrysson, & Lewandowski, 2016), acoustic emission (Lu & Wong, 2018), laser scanning, electromagnetic, radiographic, and thermographic techniques (Wuest et al., 2016). Among them, vision-based detection recently becomes popular because of its automation and reliability (Barua, Liou, Newkirk, & Sparks, 2014). The vision-based detection methods for AM may use two types of data, (i)

2D images and (ii) 3D meshes.

Methods dealing with 2D information have been developed for a long time but have some limitations. 2D images do not supply enough information for defect typologies so it is difficult to detect precise metrology of small defects, which are important to AM processes requiring high accuracy (Madrigal, Branch, Restrepo, & Mery, 2017). Most methods processing 2D information continuously take photos or record videos during AM build processes. The 2D photos or frames extracted from videos are analyzed by various imaging process techniques. Visible light cameras with infrared (IR) filters and high-speed shutters or IR light cameras are often used in this kind of AM monitoring (Douglas & Stanley, 2015). Barua et al. (Barua et al., 2014) used digital cameras with macro lens as an acquisition tool to monitor the melt pool of laser metal deposition (LMD). They took the RGB values of the images to calculate a temperature map with a linear regression model. The temperature map reflects the presence of flaws by the altered conduction of heat. Different light patterns can be used to detect defects as well. When the light from different orientations is projected on a product, a defect may disrupt the continuity of the contrast change (Shen, Li, Gu, & Chang, 2012). Another way to identify defect with 2D information is to project structured highlight on object surfaces (Nayar, Sanderson, Weiss, & Simon, 1990) (Perard & Beyerer, 1997). Usually there is a dominant reflection from specular surfaces. For instance, the structured highlight from point or striped light sources is reflected by the specular surface to form a reflectance map. If a light source is moving or multiple light sources from different directions project on the same object, local orientations of the surface can be estimated from the viewing direction, source direction and specular constraint. Extended Gaussian images (EGI), combined with the structured highlight method, can represent the 3D shape of an object surface and estimate the orientation histogram. The calculated shape features using the orientation histogram can show distribution properties of local surface orientations. It is acknowledged that this approach

for shape classification does not need registration and matching of local position features. Adopting lighted stripes, may enhance accuracy but requires greater computational effort, which is not suitable for online AM monitoring (Perard & Beyerer, 1997). Aluze et al. (Aluze, Merienne, Dumont, & Gorria, 2002) improved the computational efficiency by using both lighted and dark stripes to increase the contrast between defective and flawless areas. However, the method is only suitable to certain surfaces because the light-ray path does not reflect on certain areas of a concave shape.

Methods dealing with 3D information are arising because of the recent progress in 3D data processing techniques. Some 3D acquisition systems require multiple digital cameras at different angles to collect images and then reconstruct 3D models, while others use 3D scanners to rebuild the 3D models of parts (Lin, Shen, Fu, & Wu, 2019). Madrigal et al. (Madrigal et al., 2017) used the deformation of the light patterns to reconstruct 3D models with a descriptor called Model Points Feature Histogram (MPFH). The MPFH data are used as the inputs for a *support vector machine (SVM)*, a supervised machine learning (ML) model, to do classification. Inline Coherent Imaging (ICI) is an integrated low coherence interferometric imaging technique to reconstruct 3D models, which is closely related to spectral-domain optical coherence tomography (SD-OCT) and provides a new insight into melt pool geometry (Madrigal et al., 2017). Holzmond and Li (Holzmond & Li, 2017) proposed an *in-situ* approach in which two digital cameras are set up above the working plate from opposite directions to reconstruct 3D point clouds. A point cloud is a set of data points in space to represent the geometry of objects. The Z-differences of points in the reconstructed point cloud are calculated after registration with the Iterative Closest Point (ICP) algorithm. Registration, also known as point matching, is a process of finding a spatial transformation that aligns two point clouds. A threshold of Z-differences between two clouds is used to classify defects. Although this approach is efficient and easy to apply in production, its accuracy was not reported. In this study, we tested their method and compared it with

our method regarding accuracy. More details are provided in subsection 2.3.

In summary, most existing defect detection methods for AM processes cannot realize *in-situ* detection. Some methods can be applied *in-situ*, but they either cannot handle small-sized defects or did not report accuracy. ML algorithms are promising to increase the detection accuracy of AM defects (Madrigal et al., 2017) because ML may use multiple features simultaneously rather than only the distances between points (e.g., (Holzmond & Li, 2017)). ML has been applied in broad ranges of manufacturing (Pham & Afify, 2005) (Kwak & Kim, 2012) (Alpaydın, 2009) and demonstrated a great potential in quality control, especially in complex environments (J. A. Harding, Shahbaz, Srinivas, & Kusiak, 2006). However, ML normally requires a large data volume to train models. If all the data are from experiments, the data collection stage will be time-consuming and costly because of energy and material waste. To address this issue, this study proposes to generate a large volume of synthetic data for model training and then use the trained model for detection during real production to save time and cost.

1.2 Methodology

Real-time detection of AM defects can be realized by periodically comparing the 3D reconstructed model of an in-building part with the design mesh. The whole AM production process can be divided into multiple phases. For example, the building process of a turbine blade can be separated into four phases, as shown in Figure 1.1. Monitoring devices, such as digital cameras or 3D scanners, can collect vision data at the end of each phase. After 3D mesh of each phase is reconstructed, defects can be detected by comparing these 3D meshes with the original design mesh. If defects are detected in the third phase, as shown in the red rectangle frame in Figure 1, the AM process can be terminated to save time and materials instead of finishing the whole production.

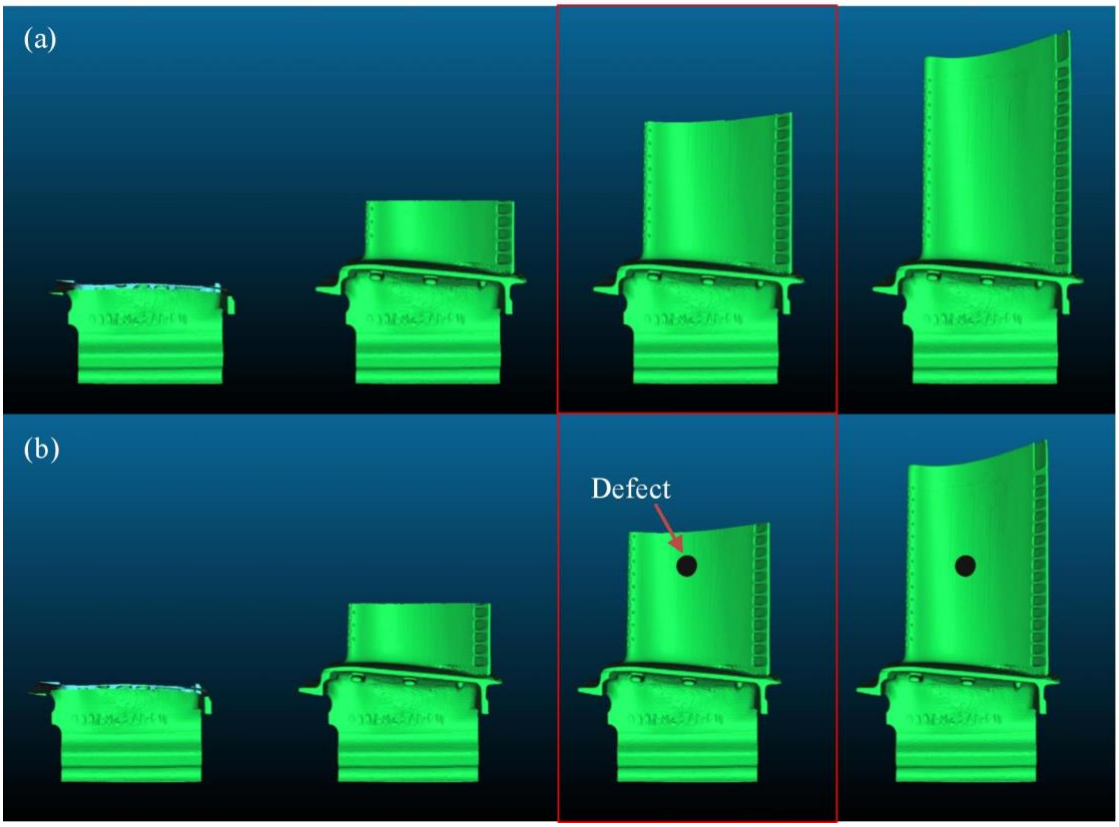


Figure 1.1 Schematic of additively manufacturing for a turbine blade with four phases: (a) perfect turbine blade; (b) turbine blade with a hemispherical defect on it.

1.2.1 Method Framework

The main framework of the proposed method is illustrated in Figure 1.2 with three steps of data generation, data preparation, and model building. For a given design, it is almost impossible to create a large number of AM builds for ML model training to detect defects due to the long build time and high costs associated with each build. Instead, we propose to use synthetic data for ML model training. A 3D mesh from the design file of an AM part is first used to generate synthetic point clouds, which include randomly created defects. The shapes of those artificial defects may be selected based on the prior experiences of defects for the same or similar AM machines and part designs. In this study, hemispherical bump defects are used to illustrate and validate the proposed method. Defective and perfect synthetic point clouds can be generated through a two-way approach that will be discussed in subsection 2.2 in detail. We use Poisson sampling and add white noise to simulate data points that are collected through 3D scanning during the implementation. The AM part point cloud without defects is referred to as a source point cloud (perfect); and the cloud with hemispherical defects is called a target point cloud (defective). Data preparation includes the registration of the source and target point clouds, the calculation of the distance and other features of each data point, which are stored in comma-separated values (CSV) for ML training. We trained five common ML models with the preprocessed datasets and evaluate them based on two criteria, F-measure and G-mean. The models and the whole proposed method were tested and compared with actual AM parts in Section 3.

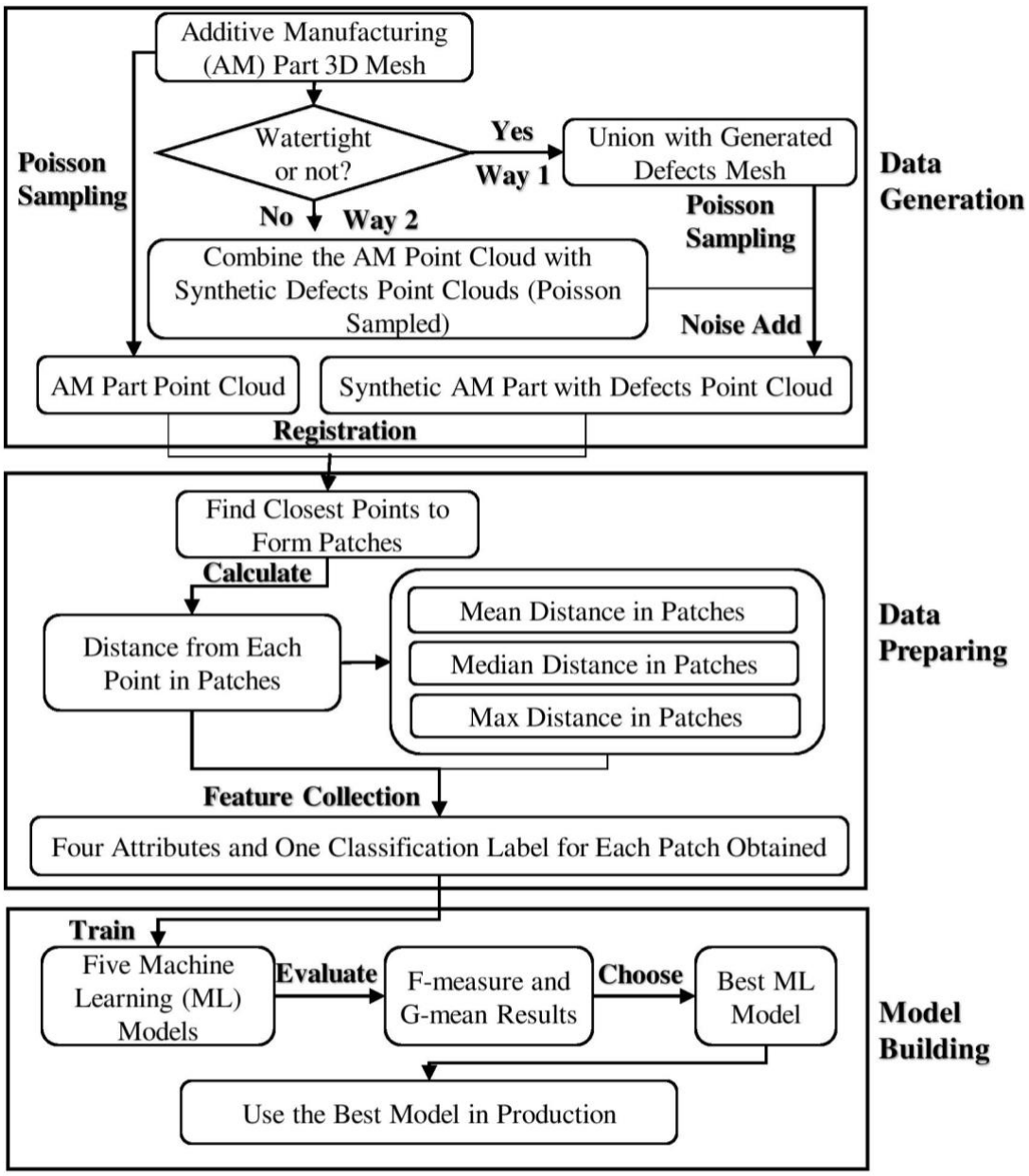


Figure 1.2 Schematic framework of the proposed method based on synthetic dataset and ML.

1.2.2 3D Synthetic Data Generation

The original 3D triangle mesh of an AM part is used as the base to add artificial defects. Defects of various shapes can be generated with python library *Open3D* (Zhou, Park, & Koltun, 2018), such as spheres, cuboids, tetrahedrons, icosahedrons, octahedrons, and torus. A hemisphere is selected in this study since it can represent common defects like bumps in AM production. After the spherical-defect triangle mesh is created, the location for the defect on the surface of the AM part is randomly chosen. Any point in the cloud converted from the triangle mesh of the AM part is a candidate of the central location of the spherical defect with the same probability.

The union of a defect and a base shape is an important part of data generation for ML training and following are two different ways (Way 1 and Way 2), depending on whether the base mesh is watertight or not. Way 1: if the base triangle mesh is watertight, like a barrel, the union can be easily realized by the *union* function in Library *Trimesh* (Dawson-Haggerty, 2019); Way 2: if the base triangle mesh is not watertight, like a flat surface, the *unio*n function of triangle meshes will not work and a manual union process is necessary. To make a hemisphere-shaped defect from a sphere and add it to the base mesh, a vector from each point on the sphere surface to the intersection between the sphere and the mesh is first calculated. Based on the sign of the dot product of this vector and the normal vector of the intersection plane, every point on the sphere can be classified into either the "internal" or "external" half. Only the external half of the sphere point cloud and the base mesh out of the intersection part, which is a plane for a flat shape, are kept and united together (Figure 1.3). The united AM part with a hemisphere-shaped defect mesh is then used to generate a 3D point cloud and Poisson disk sampling from *Open3d* is applied. Poisson disk sampling means each point has approximately the same distance to the neighboring points. It is known that in reality the reconstructed 3D models are always with some noise, so random noises $\vec{v} \cdot l$ are added on each point along a

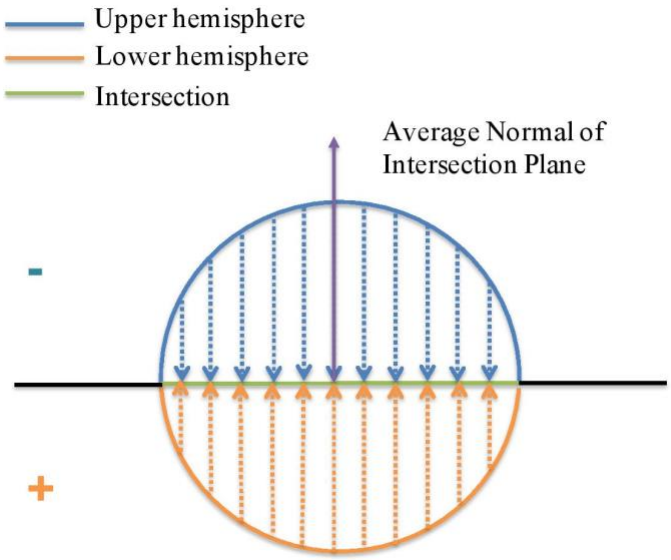


Figure 1.3 Method to generate a hemisphere out of a sphere as defect to be united with an AM part. When the dot productions of the vectors (from points on sphere to intersection plane) have the same signs, either positive (orange part) or negative (blue part), the

certain direction $\vec{v}$, where l follows a normal distribution (i.e., $l \sim N(0, \sigma)$ mm) and σ is chosen according to printing condition from experience. The parameter $\vec{v}$ here is the unit normal vector at each point.

1.2.3 Data Preparation for ML

After data generation, we have a 3D point cloud with a known defect (called source point cloud) and a base 3D point cloud without any defect (called target point cloud). We define I and J for the set of points in the source cloud and target cloud, respectively. During the training stage, we have a label $L_i \in \{0, 1\}$ for each point i in the source cloud, where "0" means "non-defective" while "1" means "defective". During the implementation stage, we are trying to classify each point i in the source cloud, which is collected through some sensors, into one of these two categories. One opportunity for AM defect detection is that we have the target cloud from a design file to facilitate this classification. The next step is data registration to match source point cloud and target point cloud to extract the features of each point in the source cloud, which will be used in ML training. During the ML implementation stage, experimental data need to be denoised before the registration. The Point-to-Point Iterative Closest Point (ICP) algorithm from python library *Open3D* (Zhou et al., 2018) and *CloudCompare* (Girardeau-Montaut, Maloney, & Janvier, 2019) are used in this study for the registration. ICP is a widely used algorithm to align two point clouds through minimizing the total difference between them. Afterwards, point-to-point distances, $d_{ij}, \forall i \in I, \forall j \in J$, between the source and target point clouds are calculated. For each point i in the source cloud, we find a point $j^*(i)$ in the target cloud with the minimum distance from i and define this minimum distance as $D_i = \min_{\forall j \in J} d_{ij}$, where $j^*(i) = \underset{\forall j \in J}{\operatorname{argmin}}\, d_{ij}$. In other words, $D_i = d_{i,j^*(i)}$. In this study, besides D_i, we will also use the information of neighboring points around point i in the source cloud for the classification because

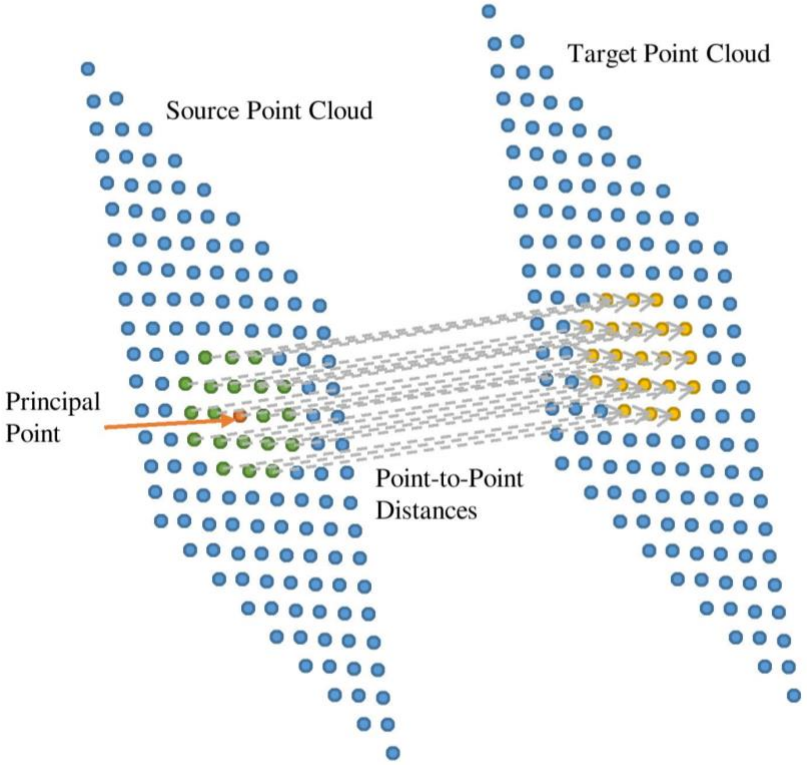


Figure 1.4 Concept of patch: Each color of dot represents one specific point in the source point cloud and each color of circle represents a sphere with points in a patch corresponding to the dot with same color.

neighboring points may be correlated due to a defect while their errors in data collection during implementation (or noises in the authentic cloud) may be rather independent.

To reflect the correlation and connection among neighboring points in the source point cloud, a concept called "patch" is introduced to obtain a more holistic relationship among the minimum distances to the target cloud of closing points in the source point cloud (Figure 1.4). For each point $i \in I$, its corresponding patch P_i is a set of n_p points in the source cloud that are the closest to point i, where $P_i \subset I$ and $i \in P_i$. Here, the patch size n_p is set at 20 point in this study and could be adjusted in practice.

Since the minimum distance of every point in the source cloud to the target cloud is known, the average, median and max distances to the target cloud among all points in a patch P_i can be calculated as follows and be used to feature the corresponding source point i.

$$E_i = \frac{\sum_{k \in P_i} D_k}{n_p}, \quad \text{Eq. (1.1)}$$

$$m_i = \underset{k \in P_i}{\text{median}}\, D_k \text{ , and} \quad \text{Eq. (1.2)}$$

$$M_i = \max_{k \in P_i} D_k. \quad \text{Eq. (1.3)}$$

The four features of D_i, E_i, m_i and M_i and the label of L_i were used for each source point during ML training.

1.2.4 Model Training and Evaluation

Five ML algorithms, Bagging of Trees (Bagging), Gradient Boosting, Random Forest, Linear Support Vector Machines (Linear SVM) and K-nearest Neighbors (KNN), were applied to build models and compared in this study. Bagging methods draw samples from data with replacement, train multiple base classifiers, and aggregate the prediction results by voting. The base classifiers are mostly decision

trees (Breiman, 1996). The first boosting algorithm is Adaptive Boosting (AdaBoosting) (Freund & Schapire, 1997), which summarizes and weights the predictions of a lot of weak learners (one-split decision trees). Gradient boosting improves AdaBoosting by optimizing an arbitrary differentiable loss function. It can adjust the decision trees depth, so the weak learners are not decision stumps anymore (Friedman, 2001). Random Forest is like an extension of Bagging. Its classifiers can choose features instead of using all features to make split at each node of the decision trees (Breiman, 2001). Support Vector Machines (SVM) can find the samples close to the boundary of different classes (support vectors). SVM finds a hyperplane that maximize the distance between support vectors and the hyperplane (Suykens & Vandewalle, 1999). Because the two classes (defective and non-defective) in the source cloud data of this chapter do not have much overlapping in the feature space, Linear SVM (Fan, Chang, Hsieh, Wang, & Lin, 2008) is employed because Linear SVM is much faster than SVMs with other kernels. KNN classifies an object through voting by its neighbors (Altman, 1991). All functions of these five ML algorithms are from python library scikit-learn (Pedregosa et al., 2011).

When the portion of the defective points in the whole cloud is too small, especially for the objects with small radius defects, the ML prediction on the rare events (i.e., defective points) can be unsatisfactory. Classifiers may predict everything as the major class. Solutions for this kind of imbalanced datasets include sampling methods and algorithmic methods (Machado, Rodrigues, Lopes, Lopes, & Mesquita, 2013). Sampling methods, like over-sampling and under-sampling, are popular solutions (Ganganwar, 2012). Adding rare class samples into original datasets and removing major class samples from original datasets are known as over-sampling and under-sampling, respectively. Under-sampling may cause loss of useful information, so Synthetic Minority Over-sampling Technique (SMOTE) (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) is first used on data in this study. SMOTE generates synthetic samples instead of replication of the original samples.

Because SMOTE demands that there should be at least six samples for the minority class. For cases with fewer than six rare samples, random over-sampling is used to make up. Both SMOTE and random over-sampling methods are called in this study from a python library Imbalanced-Learn (Lemaître, Nogueira, & Aridas, 2017).

For some cases that the above sampling methods do not work, a naïve way of sampling is applied. Since the 3D data is synthetically created, one way to increase the defect point sets is just to increase the number of hemispherical defects. This renders more balanced datasets. In order to solve the data imbalance problem, algorithmic methods are another alterative. SVM and KNN with over-sampling are supposed to be effective on predictions of minority class (Yong, 2012). Ensemble-based methods, like Bagging, Boosting and Random Forest, are also beneficial to imbalanced dataset problems (Abraham & Elrahman, 2013). Ensemble means a combination of multiple classifiers to enhance the prediction accuracy. Because of data imbalance (i.e., the defective points proportion in the whole point cloud is too small), using the simple accuracy (number of correctly classified points over number of all points) cannot effectively tell the model performance on prediction of defective points. F-measure and G-mean (Geometric mean) (Tharwat, 2018) are applied to evaluate ML models in this chapter, which are commonly used in imbalanced data classification and focus more on the minority class (Yong, 2012). From the prediction results, the numbers (points) of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) out of the classification are known, which define $Recall$, $Specificity$ and $Precision$, as shown in Table 1.1. Here, "positive" means non-defective while "negative" means defective.

Table 1.1: Definitions of Recall, Specificity and Precision

True Condition

Predicted	Positive	Negative	
Positiv	# of True Positive (TP)	# of False Positive (FP)	$Precision = \frac{TP}{TP + FP}$
Negativ	# of False Negative (FN)	# of True Negative (TN)	
	$Recall = \frac{TP}{TP + FN}$	$Specificity = \frac{TN}{TN + FP}$	

Furthermore, two metrics of *F-measure* and *G-mean* to compare ML models are determined as follows.

$$F\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall}, \text{ and} \quad \text{Eq. (1.4)}$$

$$G\text{-}mean = \sqrt{Recall \times Specificity}. \quad \text{Eq. (1.5)}$$

1.3 Results

To demonstrate the method proposed in Section 2 and evaluate its effectiveness, this study tested the method on synthetic data and conducted experiments for validation with 3D scanning data collected from actual AM-produced parts.

1.3.1 Results with Synthetic Data

First, we tested our method on a simple plane. Point clouds were generated from a 50mm × 50mm square plane triangle mesh. Hemispheres with 1.5-2mm radii were added as defects in the point clouds, as shown in Figure 1.5(a). To investigate the method performance on more complex shapes, a 3D triangle mesh of a barrel from ("3D Barrel," n.d.) was rescaled within a cubic box with 50mm as its edge length. Then it was combined with hemispherical defects that have 1.5-2mm radii to create a synthetic defective point cloud, as shown in Figure 1.5(b). When generating the point clouds, we added a random noise $\vec{v} \cdot l$ into each point, where l was drawn from a normal distribution $N(0, 0.025)$ in mm for the plane and $N(0, 0.05)$ in mm for the barrel, as introduced in subsection 2.2. These parameters are selected according to our evaluations in experiments.

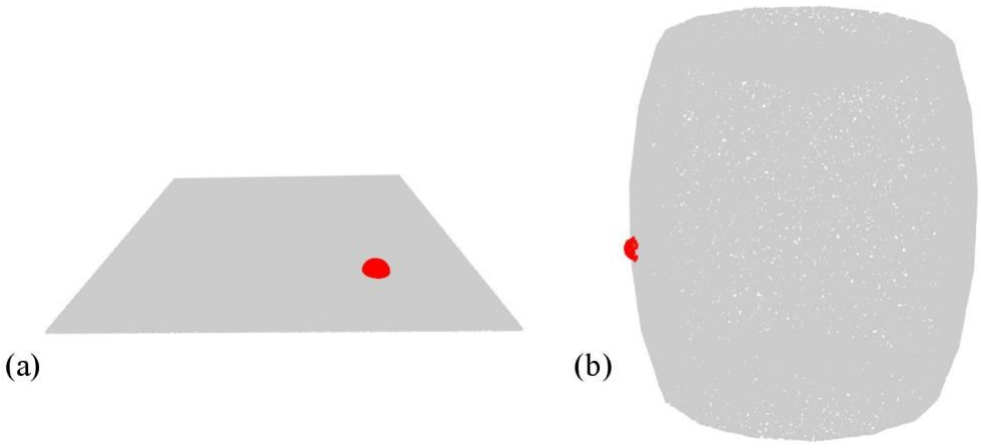


Figure 1.5 Synthetic Point Cloud of (a) square plane; (b) barrel. Red points represent defective points.

We created 50 defective point clouds with randomly created defects for each shape. Among them, 40 point clouds were used to train each of the five ML models, and the remaining 10 clouds were used for validation. To test the patch concept in subsection 2.3, we considered four sizes of 10, 20, 30, 40 points per patch. Figure 1.6 shows the *F-measure* and *G-mean* results of each ML model for the barrel with defects of a 1.5-2mm radius under different patch sizes. The figure shows clearly that the patch size of 20 points performed the best. The optimal patch size for the best ML performance should be relevant to the sizes of defects and the density of the point cloud. As a result, the patch size of 20 was used in the remaining experiments.

Five ML models (i.e., *Bagging of Trees, Gradient Boosting, Random Forest, Linear SVM* and *K-nearest Neighbors*) were trained and validated with the barrel synthetic data. As mentioned in subsection 2.4, *F-measure* and *G-mean*, rather than accuracy, were used to compare the models due to the great imbalance between the two classes. Defects with four different radii of [1, 1.5], [1.5, 2], [2, 2.5], and [2.5, 3]mm were separately added into the point clouds to see the performance of the ML models on various defect sizes. Each point cloud point had about 100,000 points and included 10 hemispherical defects. *F-measure* of each group can be determined. The *F-measure* results of the five ML models are plotted in Figure 1.7. All five models perform well with *F-measure* above 90%. Among them, *Bagging* and *Random Forest* are the best two models in predictability. Regarding computational time, *Linear SVM* is the fastest, which usually completed the calculation within only one or two minutes including both tuning and training, while the other four models took hours just for tuning.

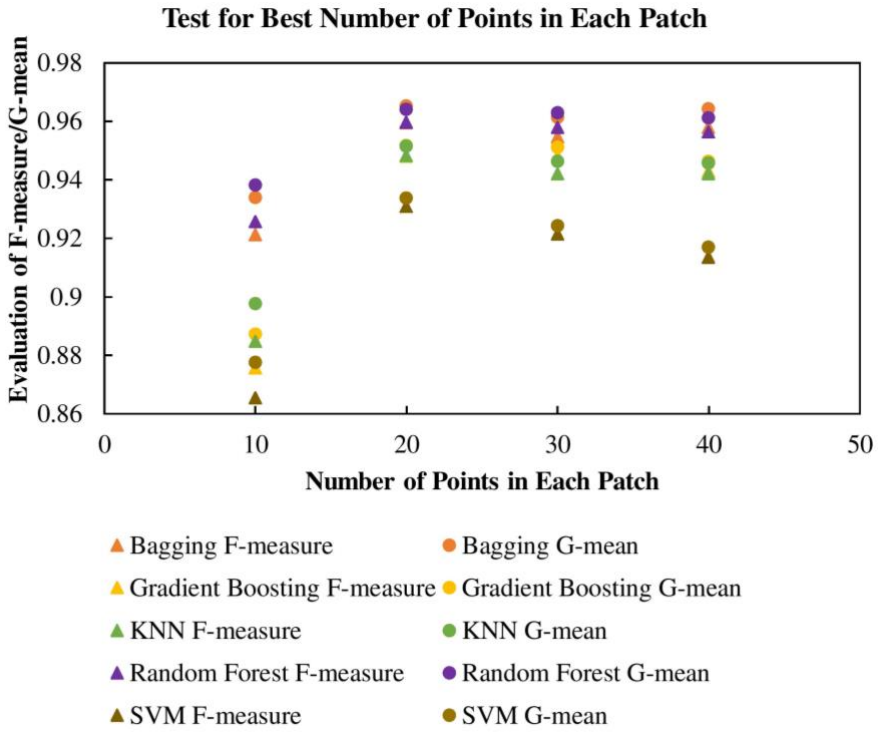


Figure 1.6 Evaluation results with synthetic data with different numbers of points in each patch

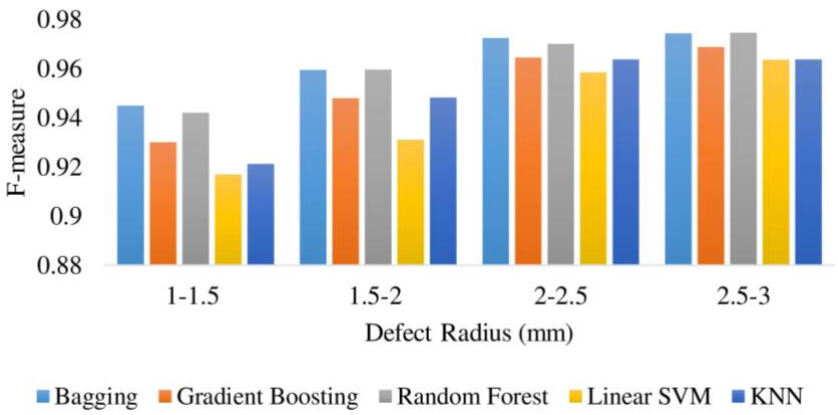


Figure 1.7 F-measure of five ML models on synthetic barrel data with different hemispherical defect sizes

1.3.2 Validation with 3D Scanned Data

In addition to using synthetic point clouds to evaluate the effectiveness of the proposed ML method, we tested the method with actual AM parts of plane and barrel shapes printed by MakerBot Replicator 2X. This 3D printer is a full-featured desktop Fused Filament Fabrication (FFF) printer using acrylonitrile butadiene styrene plastic filament with a build volume of 155×246×163mm. Figure 1.8 shows the manufactured samples of the plane and barrel with the same CAD files as synthetic data in subsection 3.1. The samples were then scanned by an EinScan SP Desktop 3D scanner. The scanned data were down-sampled into point-clouds with the same point density as in synthetic data and denoised via the software *CloudCompare*.

A great effort has been made to apply the traditional statistical method called *Z*-difference on an FFF 3D printer in (Holzmond & Li, 2017). Two cameras were set above the plate of a 3D printer to reconstruct the 3D point cloud of a surface. In their method, a threshold $t = \theta\hat{\sigma}$ was used with the *Z*-difference from reconstructed point cloud to horizontal plane to detect defects, where $\hat{\sigma}$ is the standard deviation of *Z*-difference of all points. *Z*-difference means the distance along *z*-axis from the horizontal aligned source point cloud to the target point cloud. To compare prediction results of their method and our method, $\theta = 2$ was chosen because of its best performance on our data when we compared multiple θ values. If $|Dist(x) - D| > t$, where $Dist(x)$ is the *Z*-difference at point x and D is the distance mean among all points, x is listed in the outlier set; otherwise it belongs to the inlier set. They showed that the method works well in reality, but its accuracy was not presented in their paper. For a fair comparison, the *Z*-difference method and our method with five ML models were applied to the same experimental data (one square plane and one barrel) and their performance measures of *accuracy*, *F-measure* and *G-mean* were calculated. Figures 1.9 and 1.10 illustrate the classification results for the plane and barrel, respectively, in which red points are the predicted

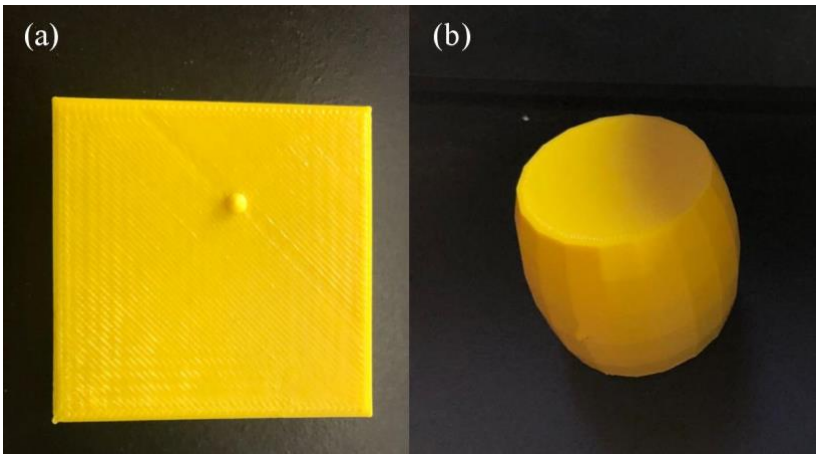


Figure 1.8 Experimental printed samples: (a) plane; (b) barrel.

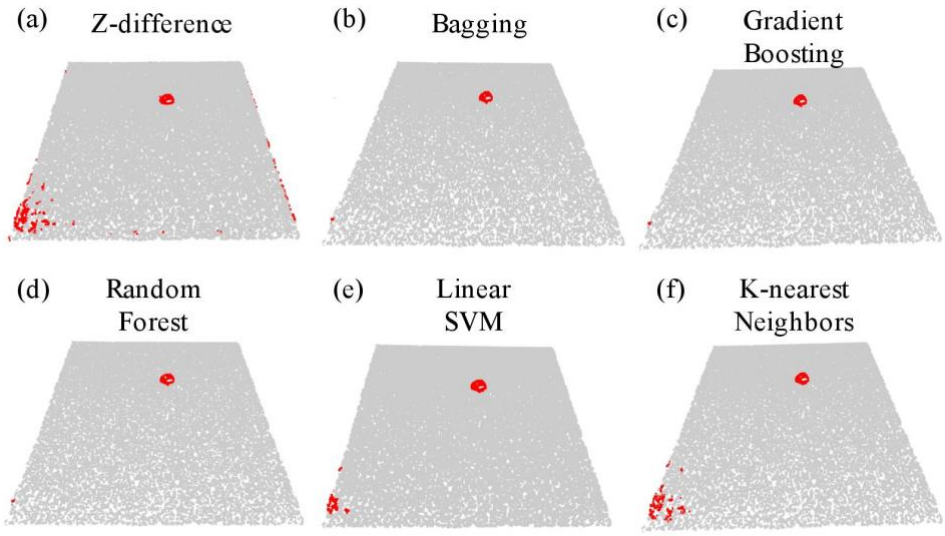


Figure 1.9 Prediction results on plane experimental data with (a) Traditional Statistical Method; (b) Bagging; (c) Gradient Boosting; (d) Random Forest; (e) Linear SVM; (f) K-nearest Neighbors.

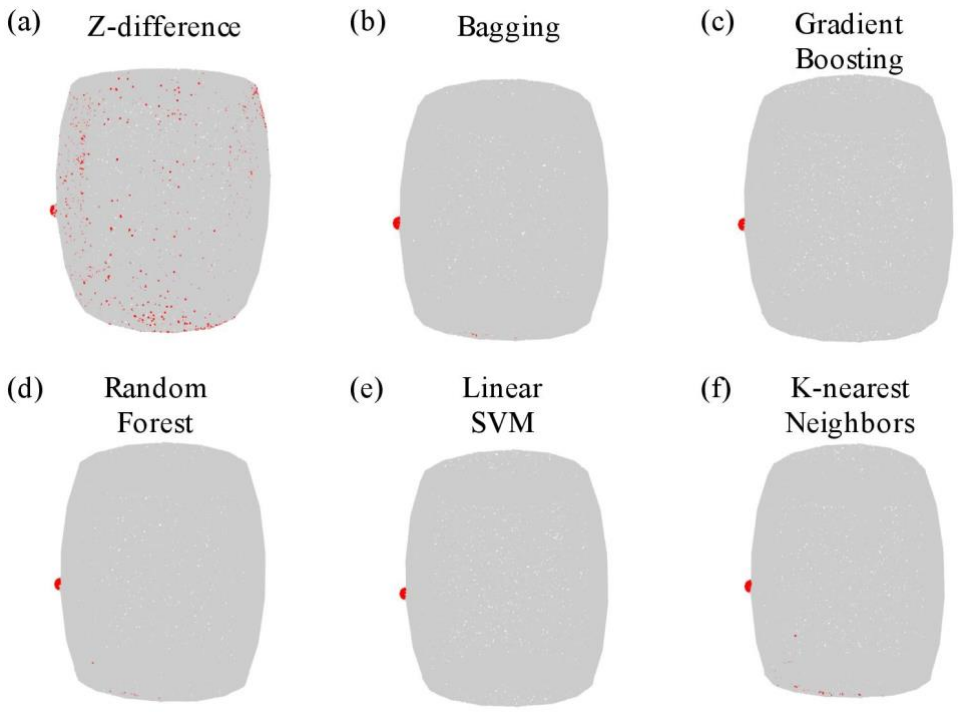


Figure 1.10 Prediction results on barrel experimental data with (a) Z-difference Method in [43]; (b-f) five ML models in this study. Red points present predicted defective points

defective points.

Tables 1.2 and 1.3 list the effectiveness of the six methods regarding the three measures on the plane and barrel, respectively. It is clear that our ML method trained with synthetic data outperforms the *Z*-difference method in detection by all three criteria, especially for complex shapes like a barrel. The $F-measure$ of the *Z*-difference method is only 5.51% while the $F-measure$ of *Gradient Boosting* is 89.08%. Different from Figure 1.5, we notice that some predicted defective points of ML models are not in the predefined hemispherical defect set. Those points were caused by actual small defects formed during printing. The plane is not exactly flat, and the bottom boundary of the barrel is not very clear. Similar to the testing results on the synthetic data, *Bagging*, *Gradient Boosting* and *Random Forest* work well on both parts. According to the $F-measure$ values of both synthetic and experimental data, *Gradient Boosting* is the best ML algorithm for detecting defects with a radius above 1.5-2mm on AM parts.

Reasons that precision of the *Z*-difference method is lower than ML methods are not difficult to conceive. A histogram of distance with regard to outliers, inliers and true defect points in the *Z*-difference method on a barrel experimental scanned data is plotted in Figure 1.10. The whole histogram in Figure 1.11(a) figure cannot clearly show the distances of outlier and defect points, so Figure 1.11(b) zooms-in and show the locations of those points at the bottom of y axis. Some outliers, which are non-defective points, are classified as defective points. There is an overlap between outliers and defective points regarding distance. Therefore, only using the distance of each individual point from source point cloud to target point cloud is not enough to effectively classify on point into the two classes of defective and non-defective, no matter how we change the threshold value. However, if more features related to nearby points are included as attributes, such as max distance, mean distance and median distance of a patch around an individual point, we increase the dimensions of

Table 1.2: Comparison of the Z-difference and Five ML Models for Detection Effectiveness on the Plane

Measure	***Z*-difference Method**	**Bagging**	**Gradient Boosting**	**Random Forest**	**Linear SVM**	**KNN**
Accuracy	0.9886	0.9978	0.9978	0.9978	0.9956	0.9926
F-measure	0.3534	0.7391	0.7391	0.7391	0.5842	0.4578
G-mean	0.4633	0.7656	0.7656	0.7656	0.6424	0.5448

Table 1.3: Comparison of the Z-difference and Five ML Models for Detection Effectiveness on the Barrel

Measure	***Z*-difference Method**	**Bagging**	**Gradient Boosting**	**Random Forest**	**Linear SVM**	**KNN**
Accuracy	0.9737	0.9992	0.9997	0.9992	0.9959	0.9979
F-measure	0.0551	0.7397	0.8908	0.7347	0.3986	0.5104
G-mean	0.1695	0.7993	0.9600	0.7946	0.5045	0.5976

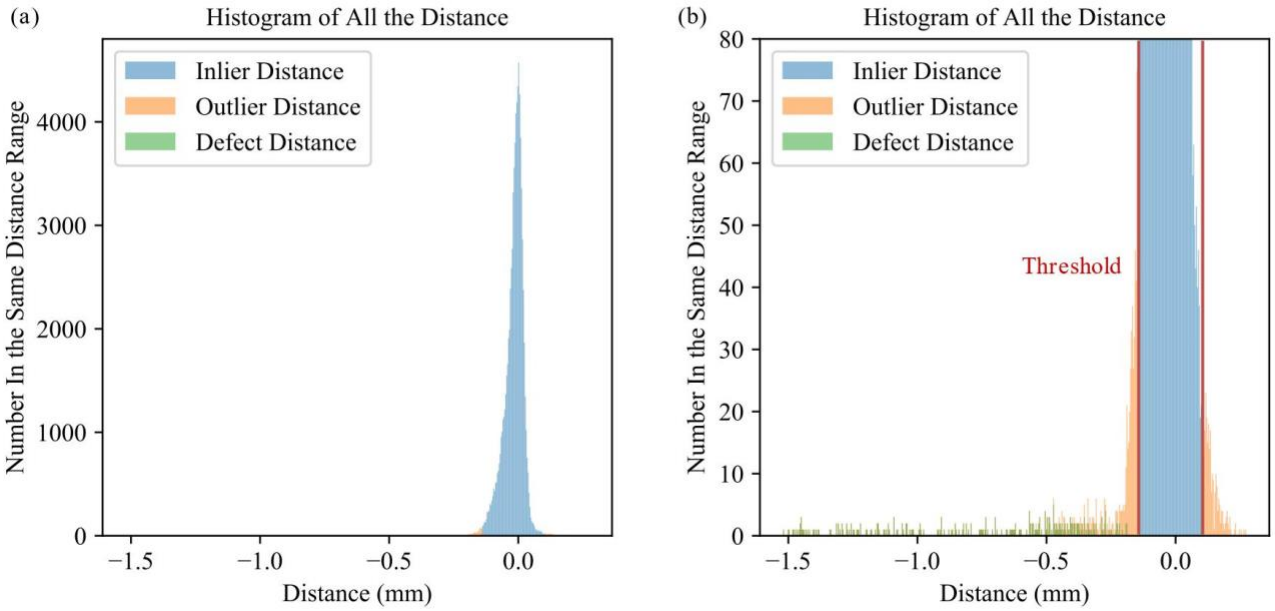


Figure 1.11 Histogram of distance with regard to outliers(orange), inliers(blue) and defective points(green) in Z-difference method on barrel experimental data: (a) whole histogram of all distance with 500 bins; (b) partial histogram with 500 bins when zooming in.

the space for classification.

1.4 Discussion and Conclusions

This chapter proposes an ML-based approach that uses synthetic data in training to defect geometric defects in AM parts. Using the synthetic data rather than actually printing and scanning many physical parts for each design for training can save huge costs and time. After incorporating information from nearby points through the new "patch" concept, the proposed method outperforms the traditional statistical method in detection accuracy. Since the ML models can be trained by synthetic data and saved in a .sav format, the trained models can be called and run quickly in real-time AM production to realize in-situ detection. This study investigated five ML models regarding $F-measure$ and $G-mean$, and demonstrated a high precision on both synthetic data and 3D scanned data from actual AM parts and outperformed the statistic method used in the literature.

Although experiments show promising results of our proposed method, its applications in real AM process still require some extra efforts. Right now, the features used for ML models are still all related to the distances between the target and source point clouds, though the "patch" concept includes some neighborhood information. More features, such as curvature, may be used to capture more macro-level features beyond individual points to improve classification precision. However, we need to address the high sensitivity of curvature calculated from point clouds. The balance between the drawbacks and benefits of having more features can be another interesting research topic. Furthermore, to realize in-situ application of our methods, an accurate real-time 3D scanner needs to be set up onto the AM machines. The AM parts were scanned actually after they were fully produced in this study. Another issue is the speed of the registration step, which took a significant amount of time in this study and will potentially hurt the training efficiency when the method is applied to many

AM designs. The proposed method trains ML model for each AM design and then use it in the production separately even though the trained models can be repeatedly used for the same design during production. We will investigate whether ML models can be trained and tested with objects of different shapes, which can significantly improve implementation efficiency. All the above research questions are worth to study and we are confident that they can all be well addressed.

This study is expected to lead to progresses in AM quality control and management. For instance, criteria or policies can be developed based on the real-time classification results to make the decisions of whether an AM production should be stopped or continue at any moment. An early termination for a defective part means huge cost and time savings for AM due to its long build time compared to conventional manufacturing. We plan to explore dynamic decision-making models, such as Markov Decisions Processes, based on the detection results in the future. Furthermore, we plan to develop a complete tool, from synthetic data generation, experimental data collection, model training, and implementation in real production. The tool should be able to generate synthetic defective point clouds for various shapes of AM parts, consider defect shapes and incorporate various geometrical accuracy tolerance requirements.

Chapter 2

Defect Detection on Additive Manufacturing Parts with Machine Learning Informed by Discrete Mean Curvature Measures

Abstract: The geometrical quality assessment for additive manufacturing (AM) is a great challenge because of the complexity of AM parts. Existing anomaly detection algorithms with 3D data mainly use features comprised of point-to-point distance difference between the designed and manufactured objects. We introduced a new curvature feature, i.e. Discrete Mean Curvature Measure, to capture macro-level information beyond the distances and incorporated it into the training data for Machine Learning (ML) algorithms. Five ML models (*Bagging of Trees, Gradient Boosting, Random Forest, Linear SVM and K-Nearest Neighbors*) were implemented on both generated synthetic and experimental data. This new curvature feature was demonstrated to well improve the defect detection performance, shown by the F-measure accuracy of as high as 94% on experimental AM barrel samples.

Keywords: Additive Manufacturing, Defect Detection, Machine Learning

2.1 Introduction

Additive Manufacturing (AM) applies layer by layer addition techniques to build products with various materials [44]. The past years have witnessed the quick expansion of AM in both industry and academy. New AM methods and materials have quickly emerged, which can shape the future life of human beings [45]. It enables individually customizing the building of products and substantially reduces lead time of manufacturing. Complex shapes that generally cannot be produced by traditional manufacturing, such as specific automotive and aerospace parts, are producible. Hence, AM products can be on market in a shorter time than traditional manufacturing (Attaran, 2017).

However, like the traditional manufacturing techniques, AM also suffers from some challenging problems. Due to the wide variety of products types, materials and designs, there is lack of comprehensive set of design principles, manufacturing guidelines and standardization of best practices (Gao et al., 2015). Researchers now mainly focus on aspects of quality control of AM, such as prediction of parameters, prediction of mechanical properties and real-time monitoring, etc. (Kim et al., 2018).

For real-time monitoring, there is a trend to use in-situ cameras and sensors to do quality analysis or monitoring in AM (Kim et al., 2018). If real-time quality assessment is realized and the defects are detected during process, time and economic costs will be saved by timely terminating or correcting. Many new defect detection algorithms have been recently developed for online or offline geometrical monitoring of AM parts by using 2D Imaging process (Lott et al., 2011)[50], temperature monitoring (such as pyrometer (Zeng, Pal, & Stucker, 2012), IR cameras (Benda, 1994), temperature sensors, etc.), computed-tomography (CT), magnetic resonance imaging (MRI) (Cohen, Laviv, Berman, Nashef, & Abu-Tair, 2009), and other signal sensors (Yadroitsev,

Krakhmalev, & Yadroitsava, 2014) as acquisition data during and after building. Besides above data types, 3D data is a good choice as input data of new defect detection methods in various fields, since it has better inspection capability. 3D scanning has the advantages of fast 3D reconstruction process and its high dimensional data with high resolution will offer more information for defect detection (Sarghie, Costea, & Liute, 2013).

One crucial step after data acquisition to the in-situ defect detection is to find methods that can reliably judge the present status of the producing AM parts, i.e., if the AM parts that have been completed so far are in good or bad conditions. There are several methods that can play this role now. A visualization algorithm is applied to represent surface roughness of 3D models with color shading of CAD images (R. I. Campbell, Martorelli, & Lee, 2002). Taguchi method is also widely used with signal to noise (S/N) ratio, analysis of variance (ANOVA), correlation, etc. for geometrical analysis (Sood, Ohdar, & Mahapatra, 2009). Sitthi-Amorn et al. used 3D scanning as the acquisition equipment to make a closed-feedback loop for AM control and evaluated the AM parts through visually comparison of aligned designed and scanned meshes (Sitthi-Amorn et al., 2015). Surface-flatness-defect detection algorithms are created and tested with scanned dense 3D point clouds. In their algorithms, horizontal flat surfaces are supposed as the reference plane and then vertical deviating distances are calculated to evaluate the roughness of planes (Tang, Akinci, & Huber, 2008). Huang et al. present a method of detecting defects with wavelet packet entropy and region normal vector features on 3D curved surfaces with 3D point clouds as well (D. Huang, Du, Li, Zhao, & Deng, 2018). Adaptive Generalized Likelihood Ratio (AGLR) approach developed by Dastoorian et al. for surface defect detection uses also 3D point cloud data (Dastoorian, Elhabashy, Tian, Wells, & Camelio, 2018).

Machine Learning (ML) algorithms are promising algorithms as general-purpose tools for many

research fields as well as in defect detection of AM parts. A general overview of this topic is referred to Wuest et al. who reviewed and summarized the advantages and challenges of ML applications in the manufacturing industry (Wuest et al., 2016). Therefore, combination of 3D data and ML come up to be a possible new method of defect detection on AM parts. One of the supervised ML algorithms, *Support Vector Machine(SVM)*, is proposed to perform real-time defect detection in AM without accuracy evaluation (Delli & Chang, 2018). A *linear SVM* classifier is applied for defect detection with 3D CT scan data and finally achieved accuracies around 80% (Gobert, Reutzel, Petrich, Nassar, & Phoha, 2018). Computer vision techniques and ML algorithms are also used to differentiate observed melt poos of Laser Powder Bed Fusion (Scime & Beuth, 2019).

In a previous study, we have constructed ML models with 3D point clouds data, but only distance features were used in the models. More effective and accurate models usually require more features. To extract the continuity of the adjacent vertices of meshes, curvature is an important feature. Since 3D meshes do not have continuous surface, discrete curvature measures are suitable choices. Both Discrete Mean Curvature Measure (DMCM) and Discrete Gaussian Curvature Measure (DGCM) were calculated on one experimental sample. The results show DMCM is a better choice here. As a result, DMCM difference of the defective and designed meshes is added as another feature to the previous distance features. After testing on experimental data, the results of ML models with this set of features are proved to outperform the ones with only distance features.

2.2 Methodology

This section will explain the DMCM feature and offer reasons of its advantages. The whole detection procedure is then described, followed by a brief introduction to ML algorithms used in this study.

2.2.1 Discrete Mean Curvature Measure

Previous studies based on 3D data did not consider the correlation among points. We argue that a defect not only causes discrepancy between corresponding points in the target and source point clouds but also lead to differences in geometric features in a larger area than a point. Curvature is a feature that can grab correlation information of neighboring points, since it intuitively means the deviations of lines from a straight line or surfaces from a flat plane. Because we have vertices of 3D meshes, discrete curvatures on each vertex are calculated. There are two commonly used discrete curvature measures, i.e., Discrete Gaussian Curvature Measure and Discreate Mean Curvature Measure. The following is a mathematical introduction of both concepts.

We denote by M a surface in the three-dimensional oriented Euclidean space R^3. We assume for simplicity that M is the boundary of some compact set V $\subset R^3$. Corresponding objects can be defined for triangulated surfaces. Assume now that V is a polyhedron with vertex set P and edge set E.

The Discrete Gaussian Curvature Measure (DGCM) of M, ϕ_v^G, is the function that associates with every (Borel) set $B \subset R^3$ the quantity:

$$\phi_v^G(B) = \sum_{p \in B \cap P} g(p) \qquad \text{Eq. (2.1)}$$

where $g(p)$ is the angle defect of M at point p, that is 2π minus the sum of angles between consecutive edges incident on p.

Similarly, since V is a polyhedron with vertex set P and edge set E, the Discrete Mean Curvature Measure (DMCM) ϕ_v^H is defined as $\phi_v^H(B) = \sum_{e \in E} length(e \cap B)\beta(e)$. $|\beta(e)|$ being the angle between the normals to the triangles of M incident on e. The sign of β(e) is chosen to be positive if e is convex and negative if it is concave.

One AM sample of a flat square surface with edge length 50mm is designed with a hemispherical

defect (Figure 2.1(a)) to show the results of DGCM and DMCM calculation. Both of them are calculated with python library Open3D.

Here is an example of calculation process of DGCM on one point [15.1552, 37.5544, 397.2720]:

First, search nearest-neighbor points around the point [15.1552, 37.5544, 397.2720] in mm within radius 2 mm. There are in total 436 points in the nearest set N.

Then, for each point p in Nearest set N, calculate the sum of angles α between adjacent triangles with p as the coherent vertex. The list of sums of angles of all points in Nearest set is [6.2834, 6.2782, 6.2816, ..., 6.2831, 6.3025, 6.2830]. The angular defect at p is then $g(p) = 2\pi - \alpha$. Similarly, the list of angular defects of all points is [-2.5327e-04, 4.9173e-03, 1.5081e-03, 3.5353e-08, -1.9327e-02, 8.7112e-05].

Finally, add all angular defects of every point in Nearest set N. Therefore, the Discrete Gaussian Curvature Measure at this point is $\sum g(p) = 62.8012$.

The map of Discrete Gaussian Curvature Measure of a plane mesh with a hemispherical defect is in Figure 2.1(b).

Here is an example of calculation process of DMCM on the same point as above:

First, find the bounds b of a space box around the point [15.1552, 37.5544, 397.2720] with a radius 2mm: [13.1552, 35.5544, 395.2720, 17.1552, 39.5544, 399.2720]. You can see it is respectively the results of coordinates of point minus radius and add radius.

Then find all the edges (candidates) lie in the bounds. There are totally 1706 edges.

For each edge e in candidates, find out the length of its part inside the bounds b. In the following, calculate the angle $\beta(e)$ between the face normal of the adjacent triangles on both sides of the edge e. The list of these angles is [0.0042, 0.0047, 0.0389, ..., 0.1392, 0.0019, 0.1010].

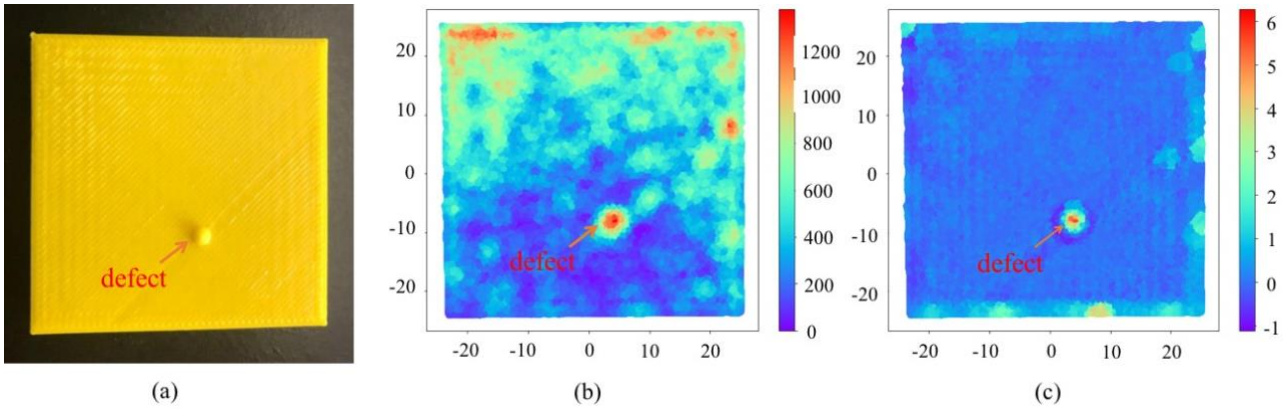


Figure 2.1 (a) Experimental defective square sample; Distribution plot of DGCM (b) and DMCM (c) on the flat square plane with a hemispherical defect. The defect is clearly shown in the plots as a red spot in the middle.

The sign is related to the convex or concave condition. What this means is that given two adjacent faces A and B, the one vertex in B that is not shared with A, projected onto the plane of A has a projection that is zero or negative, then it is convex. The sign here is obtained as +1.

Finally, the Discrete Mean Curvature Measure at this point is $\sum length(e) \cdot \beta(e) \cdot (+1)/2 = 0.0168$.

The map of Discrete Mean Curvature Measure of a plane mesh with a hemispherical defect is shown as in Figure 2.1(c).

From the above trial of the two curvature measures, values are more stable on the same plane for the mean curvature DMCM than the curvature DGCM. More importantly, it describes the defect feature outstandingly well. It is beneficial to add DMCM as an additional feature to the previous distance patch features.

Correlation analysis can reveal the significance level of each feature to describe the labels as well as the correlation between the features. To prove the DMCM is really a strong feature related to defect labels, correlation matrix of all features and the label is plotted in Figure 2.2. The figure shows that DMCM has the strongest correlation with the label. This unarguably demonstrates the importance of the curvature feature that guarantees the improvement of ML models over previous studies. Also, the correlation matrix correctly reflects the fact that the distances also have quite strong correlations with the labels. The correlations, which are slightly weaker than DMCM, are mostly frequently considered due to their simplicity in concept and calculations.

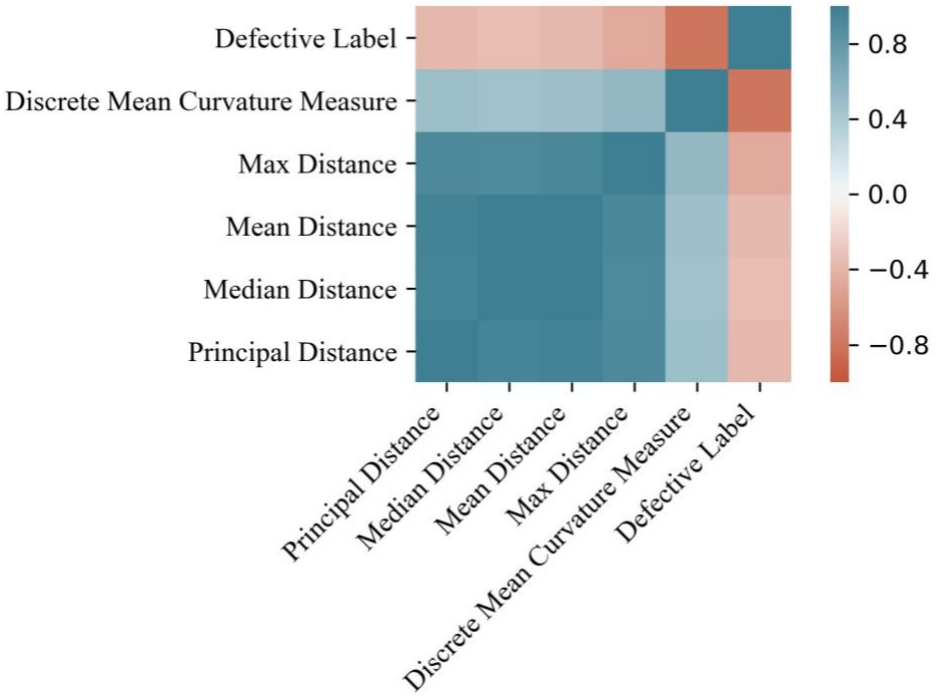


Figure 2.2 Correlation matrix of all features for ML models. DMCM has a strong correlation with defect labels.

Curvature is very sensitive to edges and pointed tips, and it can go up to infinity. For experimental data, curvatures are not strictly zero even for an assumed flat plane. The DMCM values of vertices are controlled by an important parameter, i.e., the cutoff radius of an area within which all points are considered for DMCM calculations. Given its significant influence, the parameter must be carefully tested and optimized. Albeit with the sensitivity problem, the feature is still worthy to be included as a feature in ML models due to its significant correlation with the labels.

2.2.2 Detection Framework

The defect detection framework mainly consists of three parts, i.e., data acquisition, data preprocessing and model analysis (Figure 2.3). The data acquisition consists of both generation of synthetic data and scanning of experimental data. Then these data will enter the data preprocess procedure. After registration of synthetic meshes with the designed file, and scanned meshes with the designed file, four distance features and one curvature feature are extracted. With the known labels, completed and training/validation data and testing data are prepared. The prepared data then are put into machine learning models. Five different ML models for synthesized and experimental data are trained and tested. The best ML models will be chosen according to the F-measure of test data. Finally the best results are compared with the best results of models with only distance features.

A designed barrel mesh downloaded online is used as the main base mesh and was resized into a cubic box with edge length of 50mm. There is no sufficient amount of experimental 3D data of the same sample for ML models training. Synthetic data are thus generated as training/validation data. Groups of synthetic 3D meshes with defects can be created by union or subtraction of AM objects and various small defects (such as small spherical, cuboidal or polyhedral shapes defects). Here hemispherical defects are used, since it is simplest and most common. More important, more

complicated defects can be deemed as additions of such primitive defects. The size of a hemispherical defect is controlled by the spherical radius. Radius range [1.5, 2]mm is chosen as representatives, since defects of these size well presents real defects and are also detectable by our methods. The process of generating a defective barrel mesh is shown in Figure 2.4(a). After several attempts, it is found that the defects on edges and overlapping of defects influence curvature substantially. As a result, we decided to avoid putting defects on edges as well as overlapping defects. To this end, the centers of hemispherical defects are restrained to be at least 2mm away from bottom or top surfaces of the barrel along rotational axis of a barrel. Meanwhile, the centers of multiple defects on one barrel must be at least twice the radius of the defect plus 4mm.

Data imbalance is a common problem for ML and needs to be mitigated to get an optimal model, which is also the case for our data. Compared to the whole object, the data points for defects are unproportionally small. Several methods were applied to improve the situation. One straightforward solution is, when generating the synthetic data for training, multiple defects instead of single defect are incorporated into each barrel mesh. In addition to data augmentation, algorithmic-level details can also be tailored to cope with the data imbalance. This will be introduced in next part.

As vertices of the synthetic and designed meshes are sparse, both of them are subdivided to around 10,000 vertices in each mesh. After remeshing, the meshes are able to represent details on surfaces and enough for detecting the hemispherical defects. The experimental samples (Figure 2.4 (c)) are scanned by 3D scanners. An Einscan-SP machine is used for scanning in this chapter. It has an accuracy of 0.05mm for a single scan and it can mesh to watertight data automatically. It also contains functions of smoothing and sharpening.(“‘EinScan-SP,’” n.d.) Figure 2.4(d) displays the scanned barrel mesh.

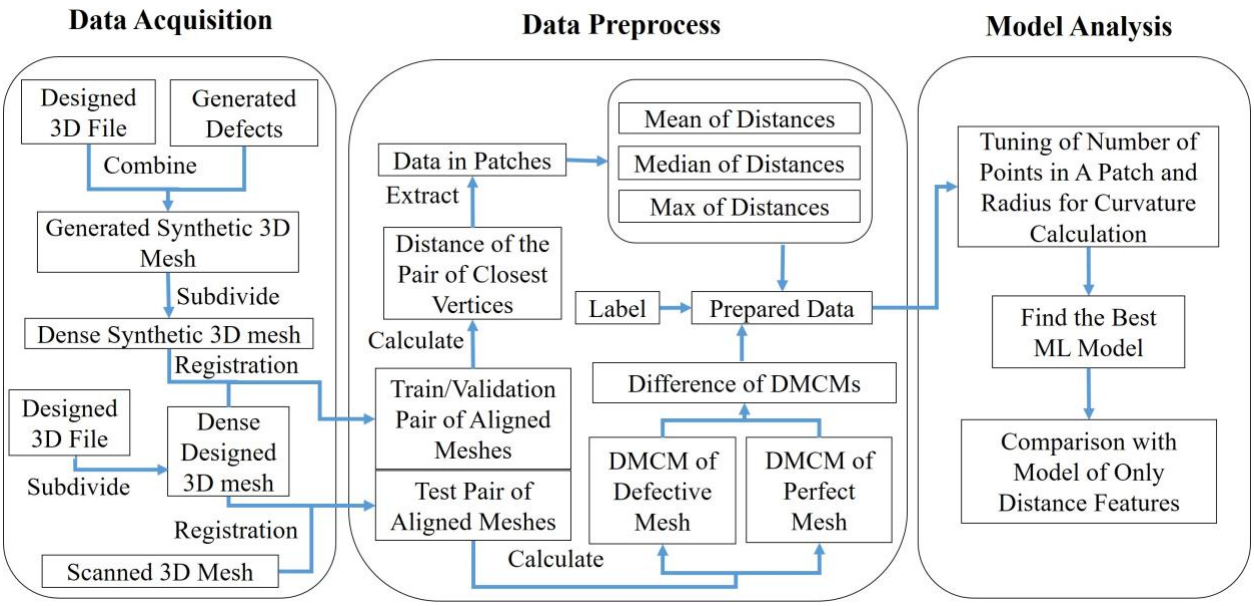


Figure 2.3 Flowchart for the automation of defect detection aided by machine learning. The process is comprised of three sections, i.e., data acquisition, data preprocessing and model analysis. Each section itself is also comprised of a flowchart with finer detail

The scanned data will again be smoothed by Taubin smooth filter to realize denoising. Therefore, in total, the whole data set is formed with three big groups of meshes corresponding to three ranges of defect radius. There are 60 meshes in total in each big group. 50 of them are with 10 synthetic hemispherical defects as training data and the rest 10 of them are with 1 synthetic hemispherical defect as validation data. One defect in each mesh of validation data is to match the experimental data, which also has only one defect per sample. Finally, mesh data in each big group is preprocessed to multiple groups of input data for ML models with various combination of the two parameters (Number of Points in Each Patch (NPEP) and Radius of Curvature Calculation (RCC)). The input data mentioned above is prepared in this preprocessing procedure. Every mesh of synthetic generated meshes and scanned meshes will go through the preprocess. The Iterative Closest Point (ICP) Registration from Open3D(Zhou et al., 2018) (point to point) is applied to the synthetic/experimental defective meshes and perfect designed meshes. Afterwards, a point to point distance, which means calculating the distance between the closest point pairs in the defective and perfect vertices, is obtained for each point in the defective mesh. Then a concept of patch is introduced here. A patch is basically a group of closest points to the picked point. In each patch, the mean, median and max of all distances are extracted as three features. To get the additional curvature measure feature, the closest vertex in perfect mesh to the vertex in defective mesh is searched. The Discrete Mean Curvature Measure (DMCM) at these two vertices are calculated and then the difference between them is obtained by subtraction of them. To illustrate the DMCM values on the barrel meshes, a threshold is set and the values beyond the threshold is marked with red color on its calculating location. Figure 2.4(b) represents the condition of target mesh and Figure 2.4(e) represents that of source mesh. The difference will be the last feature for the data as input in the ML models.

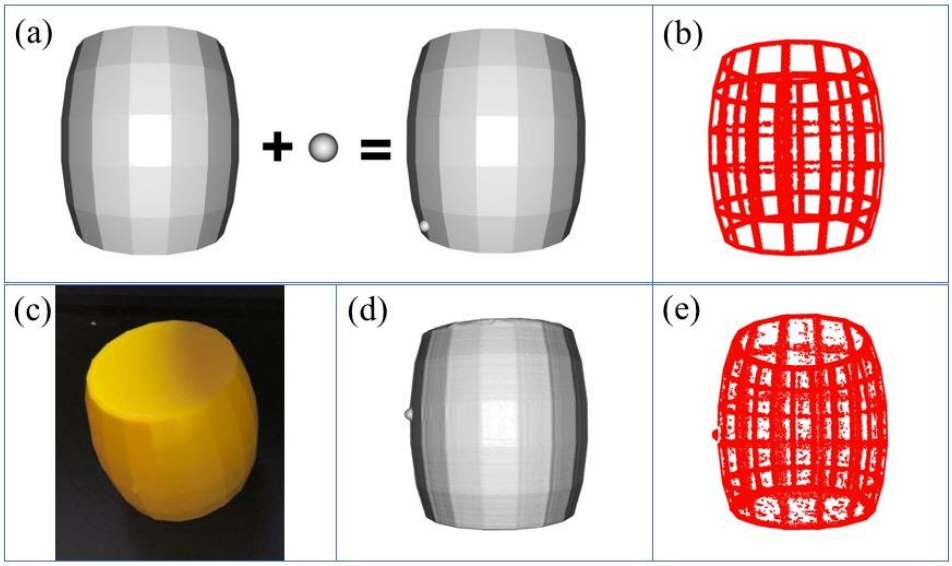


Figure 2.4 (a) Generation process of synthetic defective barrel mesh; (b) DMCM display on synthetic barrel; (c) One experimental defective barrel sample; (d) Experimental scanned defective barrel mesh; (e) Display of DMCM on experimental scanned data.

Before training the ML models, two important parameters, i.e., point density in each patch and the cutoff radius for curvature calculations, need to be tuned. Multiple groups of data with different combinations of these two parameters are generated and results of their ML models are collected in grids. Results here mean the evaluations of the models. Two types of criteria are picked to evaluate models, F-measure and G-mean. Because defects in each mesh are so small that the whole dataset is extremely imbalanced, only accuracy is unable to reflect the true model efficiency, for accuracies of all models will be extraordinarily high. F-measure and G-mean are both deduced from confusion matrix of classification results. Because F-measure is a harmonic mean and G-mean is a square root, this makes both of them closer to the smaller number, which must represent minor class. Hence, they are excellent as the role of evaluation criteria for our data and models. Mathematical forms of them are shown as below.

3D surfaces in 3rd order are fitted to these results. From the trends of 3D surfaces, the best set of the two parameters can be found. Then the selected set of parameters is used in models with five ML algorithms (*Bagging of Trees, Gradient Boosting, Random Forest, Linear Support Vector Machine and K-Nearest Neighbors*). Each model may perform a little different for each group of data, so a way to plot a figure for choosing the best algorithm is necessary. Similarly, results of each ML model are fitted with a 3D surface in 3rd order and then different colors of the surfaces are used to represent each algorithm. Finally, from the top view of the 3d surfaces, the best algorithm will emerge since their evaluation results such as F-measure will be higher. When comparing the best results of this method to the best results of models with only distance features, we could see if this method is better than the previous one.

2.2.3 Machine Learning Models

The chosen ML algorithms are all commonly used in binary classifications. Basically, five ML

algorithms are used here, i.e., *Bagging of Trees, Gradient Boosting, Random Forest, Linear Support Vector Machine and K-Nearest Neighbors*. Among them there are three ensemble algorithms (Opitz & Maclin, 1999), *Bagging of Trees, Gradient Boosting and Random Forest* respectively. An ensemble method uses a finite set of classifiers with flexible structures to improve performance of models. *Bagging of Trees* is the bootstrap aggregation of classification trees. It will train an amount of simple tree classifiers with bootstrapped data sets and then take the average output as the final output to reduce variance and helps avoid overfitting. The base classifiers of *Gradient Boosting* in this study is also classification trees. Boosting (Zhihua, 2012) can usually reduce both variance and bias through combination of weak learners, because it can adjust weightings for next learner based on the previous prediction results in the process, which means it increases the weight of misclassified data and decreases weightings of correctly classifies data. In another word, it focuses more on the data misclassified to improve the model. The main advantages of *Gradient Boosting* against other Boosting algorithms is that the classification trees in *Gradient Boosting* is with depth greater than 1 and Gradient Boosting actually optimize derivatives of loss function step by step to improve models (Breiman, 1997). *Random Forest* is quite similar to *Bagging of Trees*, but a whole set of features is applied in *Bagging of Trees*, while only a random subset of all features is used in *Random Forest*, which makes *Random Forest* have a better trade-off of bias and variance (Amit & Geman, 1997). *SVM* stands for *Support Vector Machine* and it is one of the most robust prediction methods. *SVM* (Cortes & Vladimir, 1995) creates hyperplanes with margins to classify data. *SVM* just tries to maximize the distance of the hyperplane and the data points close to it intuitively. *Linear SVM* means the kernel used in SVM is linear. It is chosen in this study for its efficiency and it is strong enough for our data. *K-Nearest Neighbors (KNN)* is a relatively simple but effective algorithm. It is instance-based learning method. For *KNN*

classifications, *k* closest points around each data point are selected and finally the class of this point is decided by these *k* points (Arian, Hariri, Mehridehnavi, Fassihi, & Ghasemi, 2020).

Another reason for applying these ML algorithms is our imbalanced data. One way to solve the imbalance problems is to use the ensemble methods. It is expected that *Bagging of Trees, Gradient Boosting* and *Random Forest* should perform well with our data (Galar, Fernández, Barrenechea, Bustince, & Herrera, 2011). Moreover, SVM is not influenced by imbalanced data, because it considers only the vectors close to the classification hyperplane (Japkowicz & Stephen, 2002). Above are the algorithmic solutions for imbalanced data. Another solution is to resample data to make it more balanced. Hence, *Synthetic Minority Over-sampling Technique (SMOTE)* (Chawla et al., 2002) and *random oversampling* (Moreo, Esuli, & Sebastiani, 2016) are applied to our data before training the models. *SMOTE* has an advantage over other oversampling algorithms, which is that it synthetized data from existing instances of the minor class, so it does not lose much information. The only disadvantage is that *SMOTE* in python library imbalanced-learn (Lemaître et al., 2017) requires at least 6 instances in the minor class. Hence, random oversampling can make it up when data does not match the requirement.

2.3 Results and Discussions

In a previous study, we constructed ML Models based on only distance features that were calculated using the Poisson sampled point clouds from 3D meshes. The best F-measure in that study is around 87% with *Gradient Boosting* models. Here we will demonstrate the results of this study with curvatures outperform the previous models.

2.3.1 Comparison of Models with/without the Curvature Feature

We apply five ML algorithms to the new data with the curvature feature added. The data is for AM barrels with hemispherical defects of radii ranging from 1.5-2.0 mm.

Table 2.1: Comparison of Models with Different Features Included on Data with Defect Radius 1.5-2mm. Here the best F-measure results are summarized.

Feature Included	**Bagging**	**Gradient Boosting**	**Random Forest**	**Linear SVM**	**KNN**
Distance	0.7391	0.7391	0.7391	0.5842	0.4578
Distance + Curvature	0.9429	0.9387	0.9432	0.9155	0.9121

The same radius range was considered in the previous study, which renders comparison of the ML models possible. The best performance of the models is recorded in terms of the F measure on the experimental data. The comparison of these results and the best results with the models trained with only distance features is tabulated in Table 2.1. As is shown in the table, the best accuracies of the five ML algorithms with the curvature feature are all above 90%, which substantially outperform the best results of previous models with only distance features. For KNN algorithm, the result is even almost twice more accurate than the previous result. This is a very persuading evidence that the performance of our new method is significantly improved. This again confirms our decision of including the curvature information is crucial, which is however missing in almost all previous studies.

2.3.2 Tuning the Number of Points in Each Patch and Radius of Curvature Calculation

We have demonstrated the crucial role of the curvature in improving ML models. Here we continue to show how to correctly calculate the curvature feature and supply the important details therein. The curvature can be tuned by two parameters, i.e., Radius for Curvature Calculating (RCC) and Number of Points in Each Patch (NPEP). Methods for choosing these parameters are proposed here. For each range of defect size, we construct a two-dimensional mesh grid with the two parameters. There are 25 grid points corresponding to various combinations of the two parameters. More specifically, we choose [1, 1.5, 2, 2.5, 3] for RCC and [10, 20, 30, 40, 50] for NPEP. In each of the 25 cases, synthetic data is collected and used to train and test a ML model. Eventually we obtain the F-measures for all the 25 ML models. Taking the F measures as the third dimension in addition to the two parameters, we plot them in a 3D scattering figure (Figure 2.5(a)). The scattered points are fitted to a curved surface represented by a polynomial up to the 3rd order. In Figure 2.5(a), each red point represents a validation test F-measure got from a group of synthetic data.

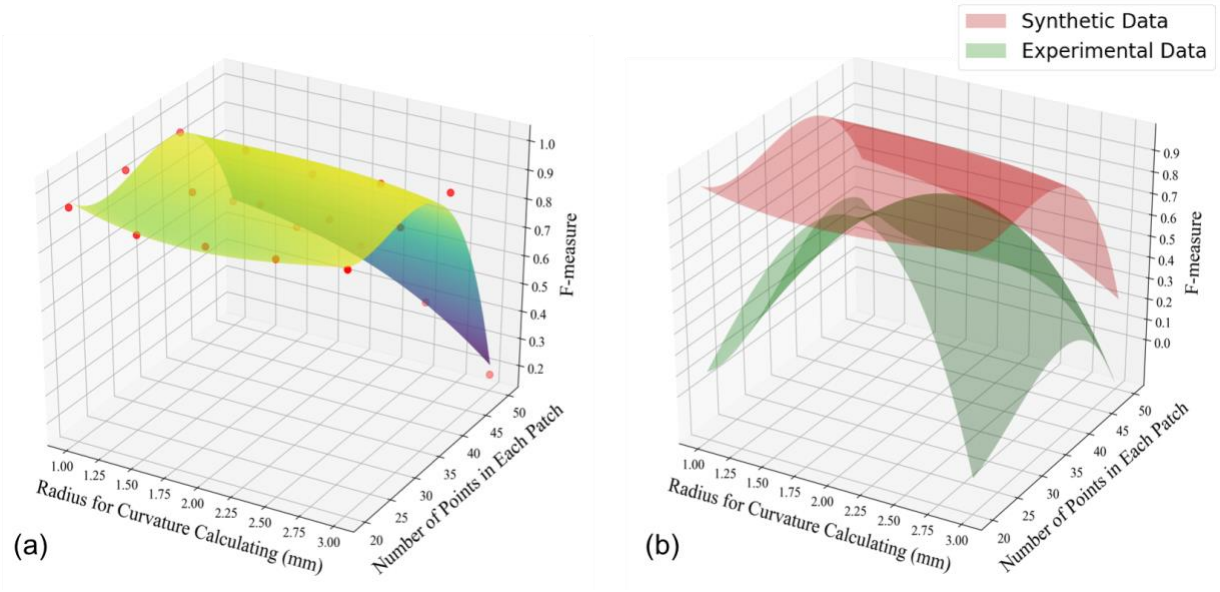


Figure 2.5 (a) 3D fitted surface of results of Gradient Boosting with synthetic data. Red spots represent real results. (b) comparison of 3D fitted surfaces of Gradient Boosting models with synthetic (Red) and experimental data (green).

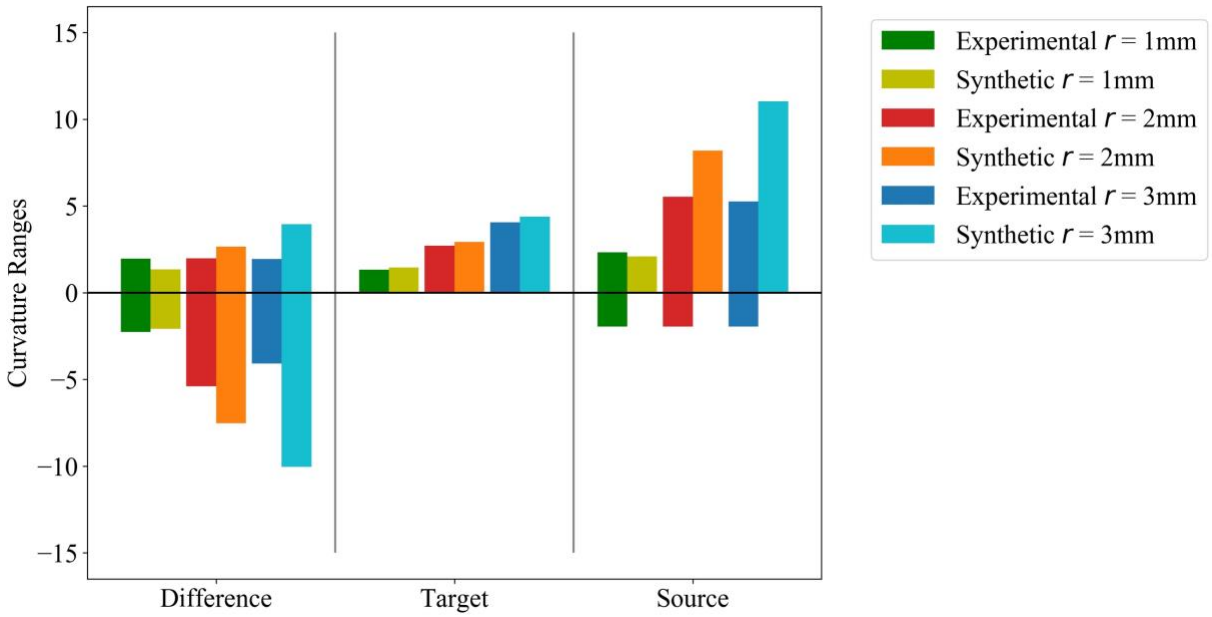


Figure 2.6 The distributed ranges of DMCM values with an RCC of 1, 2, 3mm on target mesh, source mesh and difference between them. Ranges differentiate greatly in synthetic and experimental data. Target, source and difference in the x-ticks of the figure, represen

The group of data is consisted with 50 synthetic defective meshes with 10 hemispherical defects on each mesh (training set) and 10 synthetic defective mesh with one defect on each mesh (validation set). All defects are hemispherical bumps with radii ranging from 1.5mm to 2mm. Then a Gradient Boosting model is trained with training set data and tested with the validation dataset, while there is only one defect on each mesh to imitate the real experimental data. Similarly, the trained model is also tested with data from one experimental sample (testing set). The fitted 3D surface for test results with testing set is shown in Figure 2.5(b) green layer. From the plot, we can see that the best parameter set for defects with size in this range is approximately 2mm for RCC and 35 for NPEP.

An interesting phenomenon in the results with testing set is that when RCC increases to 3mm, all F-measures of the models with different NPEP suddenly drop to 0. It seems the curvature can only be reasonably calculated in a certain range of RCC, and totally wrong curvature can be obtained outside of the range. Since this is a critical point to extract the curvature information, this question must be answered. We will explore and discuss the reasons in the next section.

When RCC is small, only a few vertices are included in the calculation of discrete mean curvature measure, which leads to fluctuate curvature values because of too small area. These curvatures are both unstable and unreliable, and more importantly, they may deviate significantly from the actual curvature. It is intuitive that when the RCC is too small, the test results are not good. To find the reasons for F-measure dropping of experimental tests with large RCC requires more exploration. To dig in, some features of Discrete Mean Curvature Measure values of different data sets are calculated and the ranges between maximum and minimum of DMCM values from source mesh, target mesh and the difference of them are plotted in Figure 2.6.

Both experimental and synthetic data are extracted for the NPEP of 50, which is taken as an

example and shown in Figure 2.6. The symbols r = 1mm, 2mm and 3mm in the legend represent the RCC of 1mm, 2mm and 3mm, respectively. The figure shows clearly that when RCC equals 3mm, the curvature ranges are significantly different between the experimental and synthetic data, which means the models trained by the synthetic data have different criterion from the models for the experimental data. It is generally accepted that the same statistic features of the datasets are the foundation of valid ML models. Otherwise, the models that perform well for the training data can easily fail for the validation data from a different distribution. While when RCC equals 1 or 2mm, the difference between them are not so big that will induce the failure mentioned above. As a result, the ML models perform well with RCC = 1mm, 2mm. This again confirms our reasoning. These results not only show the importance of correctly calculating curvatures but also offers a method to get them.

2.3.3 Best Machine Learning Model

Following the same procedure as we have done in Section 3.1, the 3D surfaces for all ML models can be obtained. When these surfaces are aligned together, the top view can be taken as a 2D map of searching best ML model with different parameters set. This map offers the ML models that best perform at the parameter space.

The Figure 2.7 demonstrates the fitted 3D surfaces of the F-measures for all the five ML algorithms as well as the overlay of five surfaces. The 3D surfaces tell a trend of the F-measure variation along with the parameters (RCC and NPEP) changing. All algorithms perform best with approximately RCC = 2mm, except *Random Forest*, which has the best performance with RCC = 2.5 mm.

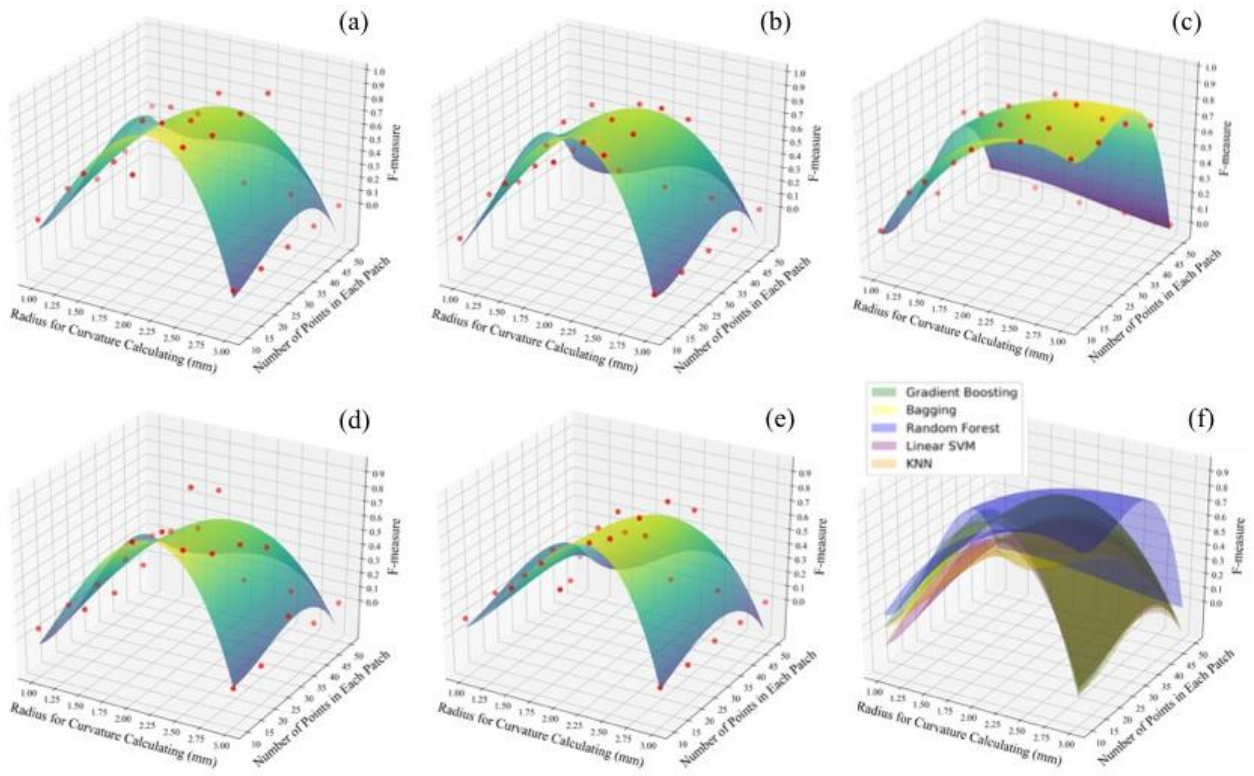


Figure 2.7 3D fitted surface of five ML algorithms. F-measure results of Gradient Boosting (a), *Bagging of Trees* (b), *Random Forest* (c), *Linear SVM* (d), *KNN* (e) and summary of five layers (f) with experimental data.

An approach of searching the best algorithm is developed through picking top values of F-measure of the five ML algorithms with various combination of the two important parameters. One example is shown in Figure 2.8. Data of this figure is barrel with defect radius 1.5-2mm. The figure shows that for this specific set of data obtains best results mainly with Random Forest algorithm. In most of the parameter space, *Random Forest* performs the best. The top view of the overlay diagram is shown in Figure 2.8. It is obvious that *Random Forest* works the best most of time, then in some situations *Gradient Boosting* works the best. Other Algorithms do not show much superiority.

2.4 Conclusions

We have proposed five highly optimized ML models for the automation of defect detections during additive manufacturing, taking barrel as an example. Both synthesized and experimental data are used in model construction and validation. The models have impressive accuracies (F-measure) as high as 94%, which outperform previous models including our own model. One of the most significant features of the models in this study is the inclusion of an additional feature that measures the local curvatures, which is missing in most of the previous studies. In comparison with previous models, we demonstrate the critical role that the new feature in improving the model accuracy. Albeit with lots of benefit, the calculations of the new feature introduce new complex to the data preparation, which if it is not correctly addressed may not benefit the ML models but even worsen their predictions. A comprehensive tuning process of the two significant parameters (the Number of Points in Each Patch and the Radius of DMCM Calculating) for the curvature calculation is conducted by a searching on the mesh grid of the parameter space. Finally, via laying results surfaces of the five ML algorithms (*Bagging of Trees, Gradient Boosting, Random Forest, Linear SVM and KNN*) one on the top of the other and then grab partial surfaces with highest F-measure to form a map of best algorithms with different parameters. The example case in this

study uses barrels with hemispherical defects and the defect radius is in range of 1.5 - 2 mm. Our results show *Random Forest* algorithm performs excellently in this case. Its outstanding performance is expected since it is an ensemble method suited for classification of unbalancing data.

There are also some limitations to our new method, due to that curvature is usually sensitive to sharp edges and points. In extreme cases when the designs of AM parts are complex and has defects on sharp edges and points, our method may suffer from serious challenges that need further efforts. Also, our method predicts defective condition on each vertex of meshes, not in local area. Some noise may be predicted as defective vertices, which can be avoided or excluded. The possible solutions to overcome the curvature sensitivity might be to divide the whole AM objects and use our method only on the parts of simple geometrical shapes. A transformation of the prediction on each vertex to a local area prediction will be created. Furthermore, a decision process, such as Markov Decision Process, will be developed for our results to decide to which level of defection the process should be stopped in the future. The combination of this study and the future planned study have potentials to realize the in-situ closed-loop quality control.

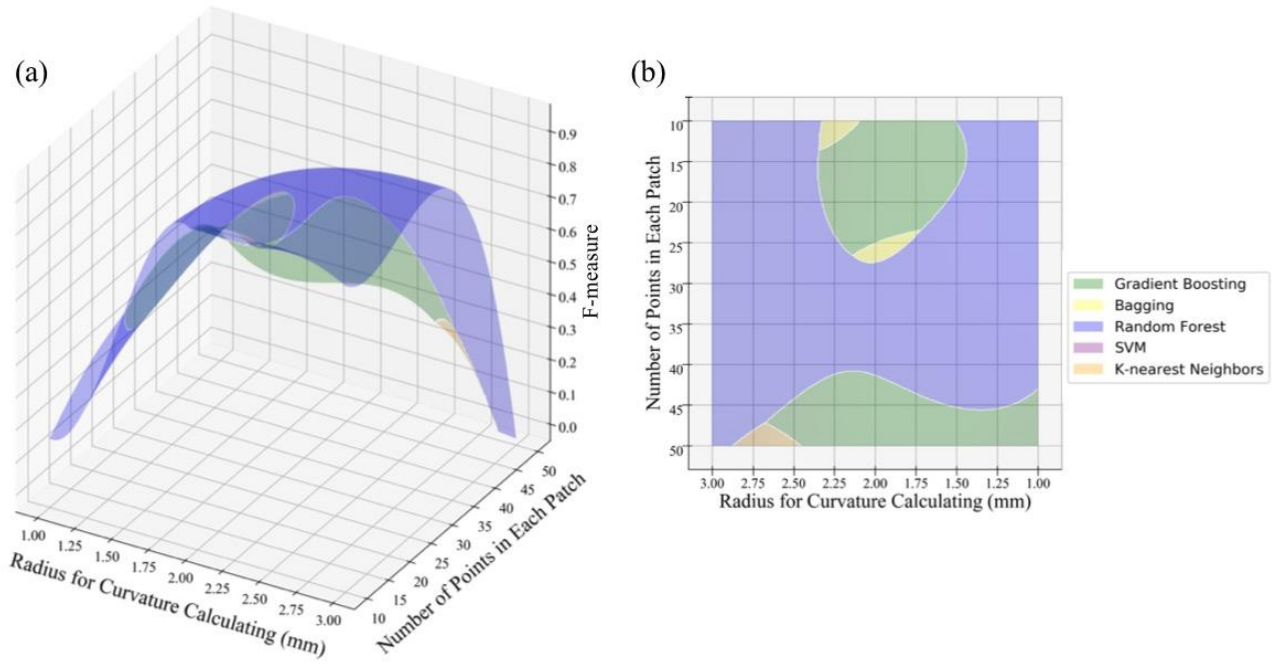


Figure 2.8 (a) Best algorithm plotting of barrel samples with defect radius 1.5-2mm. (b) 2D map of the best algorithm among five ML algorithms of barrel sample with defect radius 1.5-2mm.

Chapter 3

A Method for In-situ Defect Detection of Additive Manufacturing with Markov Decision Process

3.1 Introduction

Additive Manufacturing (AM) is a rising manufacturing technique that produces objects by a layers-by-layers manner (Huang et al., 2013). AM is able to print objects that are more complex than traditional manufacturing in shorter lead time, which includes tool design and preparation (Attaran, 2017). However, quality is a concern for AM, there is no standard guidelines or methods for its quality control and challenges appear in its real-time defect detection (Gao et al., 2015).

There are different methods for defect detection on AM parts using various data acquisition systems. 3D scanning is an attractive data-collection method with high efficiency and high resolution, since 3D data offer information of one more dimension for geometrical accuracy than 2D data (Sarghie et al., 2013). Machine Learning (ML) algorithms perform excellently in diverse fields of research and industry (Wuest et al., 2016), especially for the complex systems in which people don't fully understand relationships between physical quantities that describe them, but have a large amount of data. It is intriguing to detect defects by the combination of ML algorithms and 3D scanning data.

In previous studies we trained ML models on synthetic data and used them on real experimental data to detect the defects. Features such as distance and discrete curvature extracted from both synthetic 3D meshes and experimental 3D meshes were used in the training. The trained ML models were tested with experimental data and an accuracy 94% of F-measure was achieved. However, the ML models are applied on AM parts after the whole building process. It did not realize in-situ detection during printing process. In addition, there was no feedback and actions after defect detection in previous studies, which is the ultimate aim of this series of work. In-situ detection and real-time decision making can help to improve the productivity of AM because its build time is typically long and costly. Hence, a research plan for developing an in-situ control

system of AM is proposed here. This closed-loop on-the-fly autonomous system is comprised of applying previous ML models on segments of AM parts in a real-time fashion, analyzing on the evaluation results of the ML models, and taking actions based on analysis findings (i.e., terminating or continuing the process). Challenges of this proposed system lay in choosing the best segmentation algorithms for the 3D data, methods of analyzing results from ML models and the feedback mechanism.

Multiple algorithms for segmentation of 3D data have been developed, such as segmentation of 3D meshes through spectral clustering (Liu & Zhang, 2004), segmentation with graphs on which face nodes are constructed (Golovinskiy & Funkhouser, 2009), data driven segmentation approach with geometric and contextual label features (Kalogerakis et al., 2010). More exploration requires to select the most suitable segmentation algorithm for our study.

The optimal action policy for the feedback system is determined through the Markov Decision Process (MDP) method for the AM process. MDP is widely applied in industrial fields as well as in AM. Most of research with MDP in AM process is on supply chain. For instance, results of MDP models from Westerweel et al. (2020) showed that on-site printing is beneficial to save operational costs through on-site inventory reductions and increased asset availability. There is some MDP research about AM in-situ process control. For example, Yao et al. (2018) and Yao & Yang (2018) successfully applied an MDP model to derive the optimal control policy with defective information extracted from layer-wise image data.

Once an MDP model is constructed, the next step would be to find the solution to the model. MDP models can be possibly solved with value iteration method (Zhang & Zhang, 2001), policy iteration method (Puterman & Shin, 1978), backward value function (Satija et al., 2020), Q-learning (Watkins & Holloway, 1992) or dynamic programming (Baxter & Puterman, 1995), etc. The exact

solution method for our MDP model will be later studied and decided.

3.2 Model Framework

This research proposes three sections for a closed-loop monitor system for AM processes: processing of ML results, building of the MDP model and solving and analyzing of the MDP model. The prediction results of ML models from the previous chapters provide information about the true defective condition of the AM part, and the information is used in the state space definition of the MDP model. According to the AM process, the action space, decision policy, transition matrix, rewards and value function in an MDP model are defined. After the MDP model is built, one numerical example is made to explain details of the model. Then the model is solved, and the influence of frequency of the detection and the number of defective levels on the total value of MDP model are analyzed. A brief framework of the research is shown in Figure 3.1.

3.2.1 Machine-learning predictions as the input for MDP model

As stated above, an MDP model is proposed to be the core part of the feedback system. To build the MDP, the prediction results of ML models needs to be transferred to a defective degree, for which a defective degree reference set must be defined. An idea of the definition is shown in Figure 3.2.

Shown as above, the surface of an AM part is scanned as a 3D point cloud or 3D mesh at each stage. The whole printing process can be equally divided into N stages. For each point in the point cloud or each vertex in the printed part of the mesh, ML models will have predictions about if this point is defective or not. According to the prediction results of ML models on each point or vertex of the printed part, if N_1 points are classified as defective points and the total number of points in this patch is N_{tot}, then $N_{tot} - N_1$ points are predicted as non-defective points. In advance, there is an accuracy α of the prediction, then the proportion of defective points of ML results is

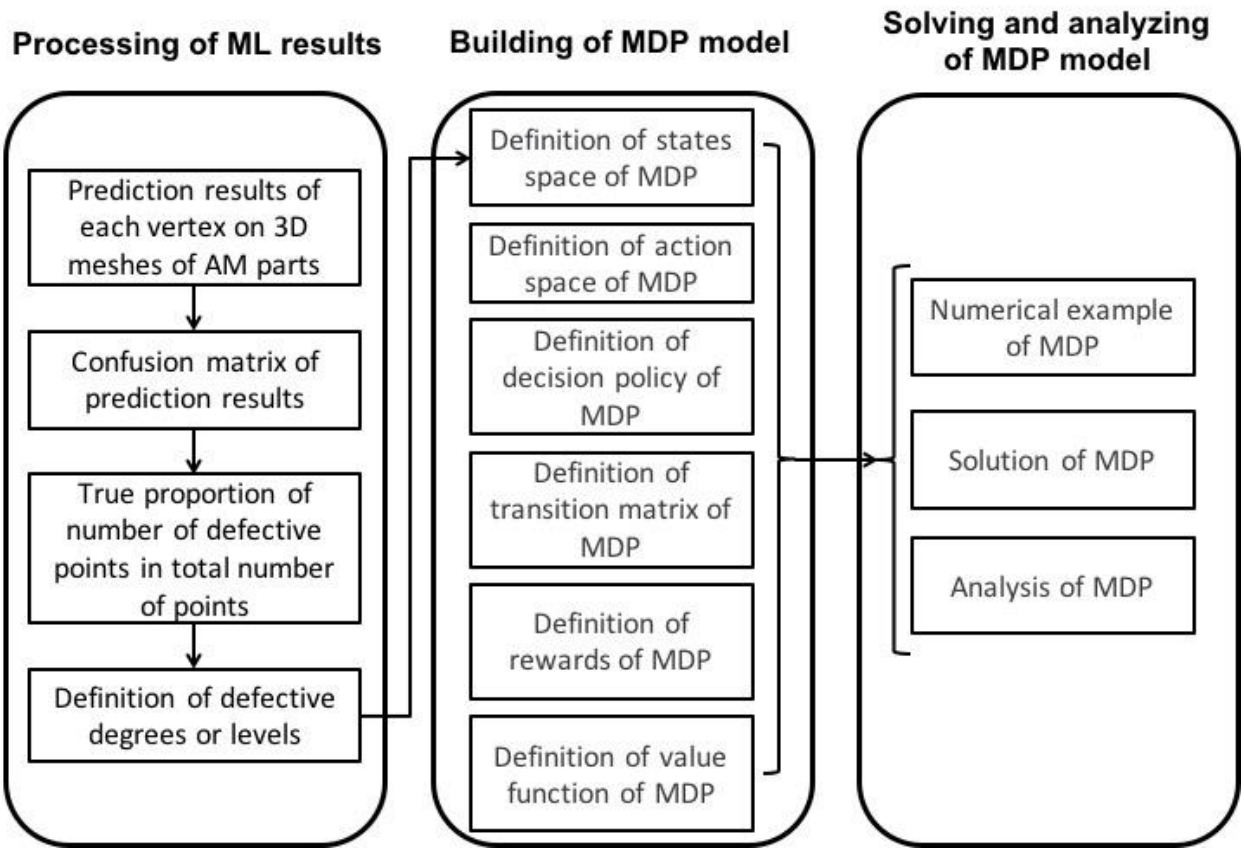


Figure 3.1 Framework of the research

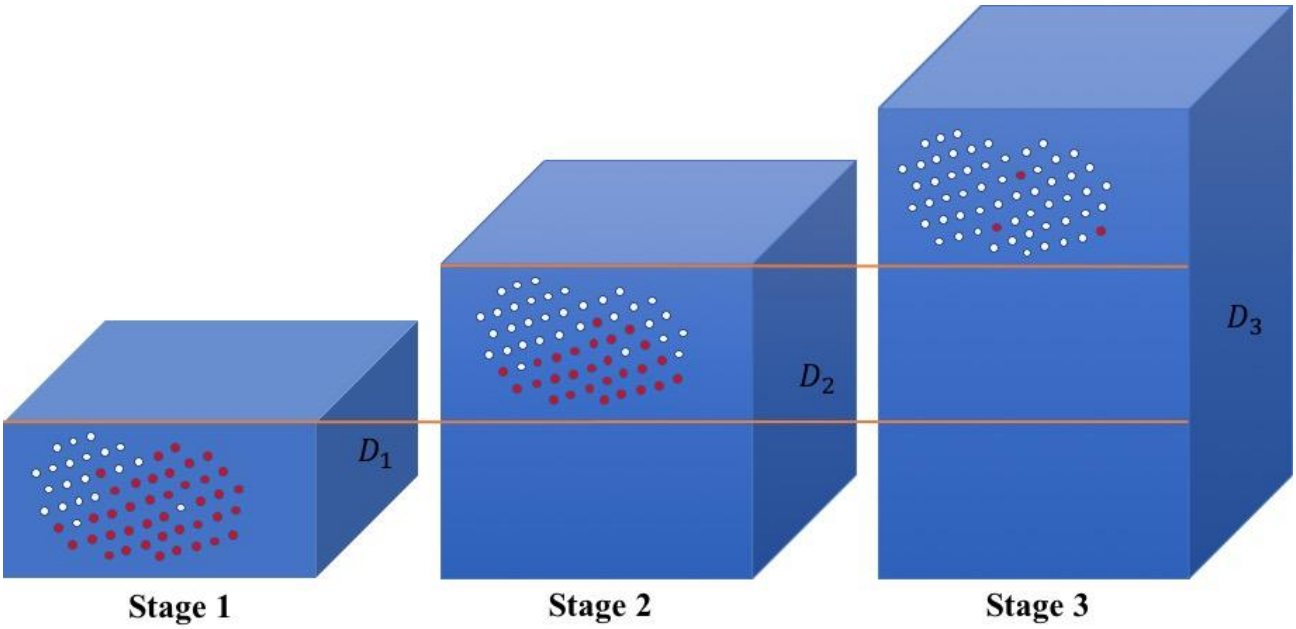


Figure 3.1 Definition of defective degree reference set. Red and white points stand for defective points and non-defective points respectively.

Table 3.1: Confusion matrix of the ML results

	Predicted Defective Points	Predicted Non-defective Points
Actual Defective Points	$N_1 \cdot \alpha$ (TP)	$(N_{tot} - N_1) \cdot (1 - \alpha)$ (FN).
Actual Non-defective Points	$N_1 \cdot (1 - \alpha)$ (FP)	$(N_{tot} - N_1) \cdot \alpha$ (TN)

$$\rho_{ML} = \frac{N_1}{N_{tot}}. \quad \text{Eq. (3.1)}$$

Accordingly, the number of true positive points is $TP = N_1 \cdot \alpha$ and the true negative is $TN = (N_{tot} - N_1) \cdot \alpha$; the number of false positive points is $FP = N_1 \cdot (1 - \alpha)$ and the false negative is $FN = (N_{tot} - N_1) \cdot (1 - \alpha)$, shown as in the following Table 3.1.

From above we can see that the expected number of actual defective points is estimated at

$$\begin{aligned} N_d &= N_1 \cdot \alpha + (N_{tot} - N_1) \cdot (1 - \alpha) \\ &= N_1 \cdot \alpha + N_{tot}(1 - \alpha) - N_1 \cdot (1 - \alpha) \\ &= N_1(2\alpha - 1) + N_{tot}(1 - \alpha) \end{aligned} \quad \text{Eq. (3.2)}$$

Then the expected proportion of the defective points in the total points is considered

$$\begin{aligned} \rho = \frac{N_d}{N_{tot}} &= \frac{N_1(2\alpha - 1) + N_{tot}(1 - \alpha)}{N_{tot}} \\ &= \frac{N_1}{N_{tot}}(2\alpha - 1) + (1 - \alpha) \\ &= \rho_{ML}(2\alpha - 1) + (1 - \alpha) \end{aligned} \quad \text{Eq. (3.3)}$$

When ML accuracy $\alpha = 100\%$, the true proportion of defective points $\rho = \rho_{ML}(2 \times 100\% - 1) + (1 - 100\%) = \rho_{ML}$, which is consistent with the case with a full confidence in ML. In other case, there is always some deviation between the two quantities. For example, assuming $\rho_{ML} = 5\%, \alpha = 99\%$, we can calculate $\rho = 5.9\%$, slightly larger than ML prediction of 5%. At each stage, if the proportion of the defective points in this stage is ρ_n, then the defective degree of the printed part can be defined as

$$d_n^* = \rho_n, \quad \text{Eq. (3.4)}$$

3.2.2 Definition of MDP model

To make the states infinite and discrete in our MDP model, the state value can be rounded off to

certain number of decimals as

$$d_n \approx [d_n^*], \qquad \text{Eq. (3.5)}$$

A series of discrete defective degrees form a reference set $\{d_1, d_2, d_3, \dots, d_M\}$ to represent the defective condition of AM parts and the number of index reflect the level of defective degree in an ascending order. With the set of defective degrees as one part of state space, an MDP model might be defined as below.

A process diagram of an example is plotted in Figure 3.3. We suppose the whole AM process is observed in N stages with equally divided time epochs. In each stage, there are possibly different state conditions on the defective degrees of the newly added part. Another assumption is that at each state, there is no correction action to improve defective condition. It is supposed that the defects of AM part are formed gradually rather than instantaneously.

Another interesting question here is what the probability for a part is going to a worse condition over two printing stages. There can be two possible patterns. One is to consider the AM machines working steadily and the quality at each stage is independent from that at other stages, so the probability of condition transferring is an empirical distribution, for example a normal distribution, which needs experiments to test out. Another type is to consider the quality (state) at each stage is not totally independent and it may depend on the last state, the probability distribution of state transferring will change at each state and the details of change will be discussed later in the following sections. Comparing the MDP models with these two different types of state transition distributions can be one good topic as well. The following shows the flowchart of MDP model in this research, which considers the first assumption stated above.

The following is the definition of each part of the MDP model in details.

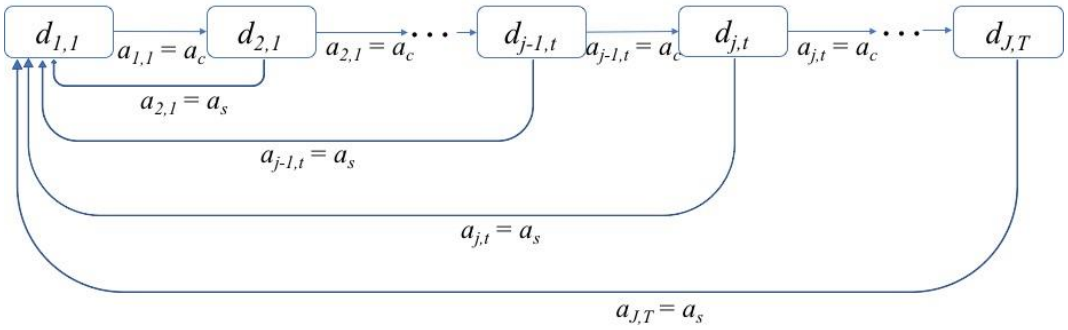


Figure 3.2 An example of the schematic for the Markov Decision Process. d represent the defective degree. The stage is a key parameter to denote the completeness of the process.

1) State Space

The state space for this process is a composite of two indices. One is the defective degree and the other one is the stage index or the observation time of the building process of a whole part $s = (j, t)$, $t \in \{1, \dots, T\}$. We suppose there are x defective levels in the first epoch or first stage, then $j \in \{1, 2, 3, \dots, x + (t-1)(x-1) = tx - t + 1\}$

$$S = \{(1,1), (2,1), (3,1), \dots, (x,1), (1,2), (2,2), (3,2) \dots, (2x - 1,2), (1,3), \dots, (1,t), (2,t), \dots (j,t), \dots, (tx - t + 1, t), \dots, (J,T)\}, \quad \text{Eq. (3.6)}$$

is what we defined previously to represent state space. In this set, j is the integer index of the defective level at the observing time epoch t. $d_{j,t}$ is the defect level at state (j, t) accordingly. Small index number of j means low defect degree, e.g., $d_{(1,t)}$ is the lowest defect degree, and $d_{(1,t)} = 0$, and $d_{(J,t)}$ is the highest defect degree at stage t. The time vector in S is a set of continuously ascending integers, $\{1, 2, 3, \dots, T\}$. Since we suppose the quality between two different stages are independent, then at the second stage or observation time epoch, defective levels $d_{1,2}$ and $d_{2,2}$ can be deduced as $d_{1,2} = \frac{d_{1,1}+d_{1,1}}{2}$, and $d_{2,2} = \frac{d_{1,1}+d_{2,1}}{2}$, and so on, because the volume in the second stage is twice as much as the volume in the first stage. Similarly, at the third stage, defective levels $d_{1,3}$ and $d_{3,2}$ is deduced through $d_{1,3} = \frac{2d_{1,2}+d_{1,1}}{3}$, and $d_{2,3} = \frac{2d_{1,2}+d_{2,1}}{3}$, and so on. The rest can be done in the same manner. Suppose $S = \{s_1, s_2, \dots, s_n, \dots\}$ is a discrete-time Markov Decision Process. T is the total number of decision epochs or decision stages and J is the index of highest defective level.

In practice, the defective degrees will be wisely chosen from the predictions of the ML model using Eq. (1). As an initial step, we suppose three different defect levels at the first stage, i.e.,

$$\{0\%, 4.5\%, 9\%\}. \quad \text{Eq. (3.7)}$$

These numbers can be calculated from experimental data based on Eq. (3.1). When the ratio of defective points is larger than 9%, they are all labeled as 9%, because this situation occurs rarely. Thus, the states of this epoch will be $\{(1,1),(2,1),(3,1)\}$. In the next step, since the volume of the printed part will be doubled, then the ratio of defective points in this step will be $\{0\%, 2.25\%, 4.5\%, 6.75\%, 9\%\}$ and the possible states set at this step will be $\{(1,2),(2,2),$ $(3,2),(4,2),(5,2)\}$. The number of defective levels will continuously increase as the printing time goes. We realize that the ratio of defective points over the whole printed points is usually very small, which underestimates the actual influence of the defect. More seriously, a very small defect level results in a trivial solution to MDP model, i.e., the optimal policy is always printing. Therefore, we magnify the number of defect points by one order of magnitude. Such a choice is not unique and will be further adjusted until a reasonable decision policy is obtained.

2) Action Space

The action space of this process is defined with two actions,

$$A = \{a_s, a_c\}. \quad \text{Eq. (3.8)}$$

Here, a_s means the defective degrees of printing part is above the defect threshold d_ξ and the process is stopped, and a_c represents the action of continuing printing. For convenience, we use numbers to represent the action space in simulation, i.e.,

$$A = \{0, 1\}, \quad \text{Eq. (3.9)}$$

Where $a_s = 0, a_c = 1$.

3) Decision Policy

The policy of this process is the action taken given the state (j, t),

$$\pi(j, t) = a_{j,t}. \quad \text{Eq. (3.10)}$$

The optimal policy π^* needs to be obtained through optimization.

4) State Transition

The key property of MDP is its nearsightedness, i.e., the current step is completely determined by its preceding step. These again can be classified into two situations. The first situation is the preceding step indeed affects the probability of its following step, where a joint probability between these two steps is needed. This requires detailed experimental data to determine the specific value. In an extreme and simpler situation, we can assume each step is independent and determined by a certain distribution that can reflect the actual situation, such as a Gaussian distribution.

Let

$$P(d_{j',t+1}|d_{j,t},a) = P_{(j,t),(j',t+1)}(a) \qquad \text{Eq. (3.11)}$$

be the one-step transition probability of this Markov chain from defective degree state $d_{j,t}$ of stage t to defective degree $d_{j',t+1}$ of stage , $t+1$ under a given action a. In practice, the occurrence of different defective levels at one stage should obey some distribution. This has to be obtained by results of experiments. We assume that the distribution of the probabilities in the first stage obey a normal distribution. As stated in the definition of state space, if we consider the highest defective level is 9%, then distribution of the probabilities is assumed as $N(0.045, 0.01)$, shown as in Figure 3.4.

Defective levels lie outside [0%,9%] is truncated and the distribution of probabilities in between [0%, 9%] is normalized. When it is decided that there are x defective levels in the first stage, then the whole distribution will be uniformly divided into x ranges from 0% to 9% defective levels. Then the probability for each defective level is deducted through integration of the area under the curve in its range.

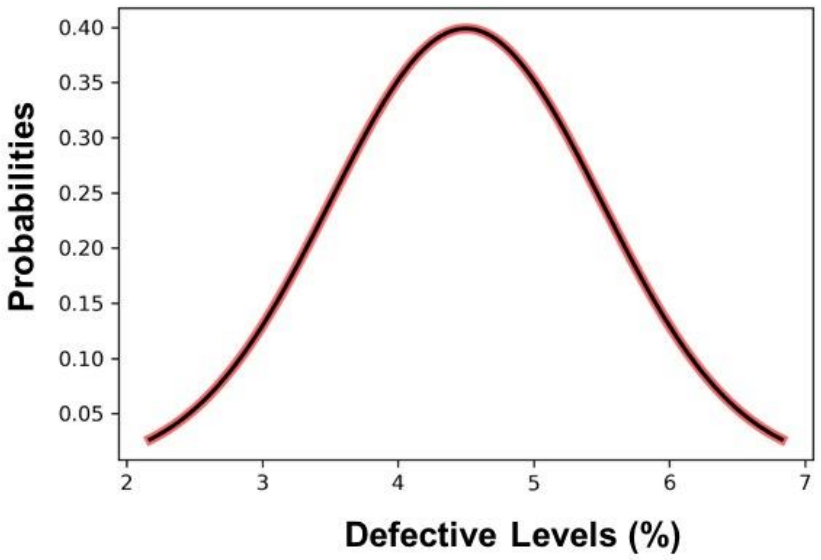


Figure 3.3 Distribution of probabilities at first stage

If one of the divided ranges has an upper bound x_{upper} and lower bound x_{lower}, its probability will be $p_x = \int_{x_{lower}}^{x_{upper}} pdf(x)dx$ and its corresponding defective level will be $\frac{\int_{x_{lower}}^{x_{upper}} x \cdot pdf(x)dx}{\int_{x_{lower}}^{x_{upper}} pdf(x)dx}$.We define the probabilities of the appearance of each defective level at the first stage as this set $p = \{p_1, p_2, \dots p_x\}$, where $p_1 = P(d_{1,1}, a_c) = P(d_{1,1}, a_s)$ and x is the number of the defect levels in first stage ($x = 3$ in the example). Then the probability in the later stages will be related to these probabilities, their defective levels and the stage they stay. Let's consider the case of a_c, which means the action is to continue printing. There are many ways to form the transition matrix, for example, supposing at each stage, the probability keeps the same as in the last previous stage, since we can assume each layer is independent. Or another idea is that when the number of stage increases, the probability of low defective levels occurrence will decrease, which means as the printing process goes on, the probability of printing defects increases. However, because now we are still in planning period, it is better to start with the simplest idea. Thus, the first idea is adopted here, which means in the second stage, $P_{(2,1),(1,1)}(a_c) = p_1$. In general,

$$P_{(j,t),(j',t')}(a_c) = \begin{cases} p_{j'} & j' \in \{1, \dots, x\}; t' = (t \ mod \ T) + 1 \\ 0 & otherwise \end{cases} \quad \text{Eq. (3.12)}$$

When the action is a_S, which means to stop the process, then all states at other stages will return to the stages at first stage and the probability will be the same as at the beginning we assumed, since it is just a new printing process starting, then right now

$$P_{(j,t),(j',t')}(a_s) = \begin{cases} p_{j'} & j' \in \{1, \dots, x\}; t' = 1 \\ 0 & otherwise \end{cases} \quad \text{Eq. (3.13)}$$

5) Reward

The reward of this process contains mainly two parts, the penalty and the gain. Assuming as above

that printing a whole AM part requires time T, at each stage, the time of the printing should be $\frac{1}{T}$. If the material cost of a whole AM part is p_0 and the value of a section of printed part is $\gamma \cdot p_0$, then the reward of the process can be defined as:

i) When $t = T$

$$R((j,T),a_c) = R((j,T),a_s) = \underbrace{-\tau \cdot d_{j,T} \cdot p_0}_{\text{Cost for post processing}} \overbrace{-\frac{p_0}{T}}^{\text{Cost of material}} + \underbrace{\gamma \cdot p_0}_{\text{Gain of printed section}} \qquad \text{Eq. (3.14)}$$

where the first term is the cost for post processing, the second for material cost and the last term for the gain of printed AM part.

ii) When $t \neq T$

$$R((j,t),a_c) = R((j,t),a_s) = -\frac{p_0}{T} + \psi \cdot C_s, \qquad \text{Eq. (3.15)}$$

Here we use the parameter γ ($\gamma > 1$) to show the increased value of the printed defect-free part and also τ to show the ratio of cost proportional to defect levels. C_s represents save of the cost if the monitoring process is applied and ψ is a ratio of the cost save regarding different situations. A value of 1 or less leads to a trivial solution to the MDP model, i.e., never printing, because there is no benefit to encourage taking the failure risk. The increased value depends on the printed parts, which we perform a parametric study later on.

6) Value Function

With all above definitions and assumptions, the optimality equation for the value function of the MDP can be expressed as below

$$V(j,t) = \max_{a \in \{a_c, a_s\}} \left\{ R((j,t),a) + \sum_{(j',t') \in S} P_{(j,t),(j',t')}(a) V(j',t') \right\} \quad \text{Eq. (3.16)}$$

Developing and improving of the integrated MDP model still need more research work as well.

7) Solution

There are several methods to solve MDP model. In this study, we firstly tried Monte Carlo simulation to solve one example model in the following. At each Monte-Carlo step, a set of actions are randomly proposed $\{[a_i]\}$ and its corresponding value is calculated using Eq. (3.16). All action proposals and their value functions are recorded and compared, the actions with the best value function is considered to be the best policy. After a sufficient long Monte-Carlo time, e.g., 1,000 steps, the best policy and value function become converged and do not change. Value Iteration and Policy Iteration are also implemented as solving methods.

3.3 Numerical Experiment

3.3.1 Numerical Example Description

We consider showing a simple case to demonstrate our research ideas. The example contains the most essential components of the MDP model albeit simple. It is assumed that there are three observing timing during the printing of a whole part, the end of the 1st, 2nd and 3rd stage. As stated above, if we consider there are three possible defective levels in the first stage of the process $\{0\%, 4.5\%, 9\%\}$. Then the schematic figure of the example will be like in Figure 3.5. Later on, we will add more actual details to the model and adjust it to our AM case.

Problem Definition. To show the rationality of our model, we tried to solve a simple example MDP of our idea. It is supposed and deduced that the states of this example MDP is shown above as in Figure 3.5, $s = (j,t)$; $t \in \{1, \dots, T\}$, $j \in \{1, 2, 3, \dots, x + (t-1)(x-1) = tx - t + 1\}$. Since here $T = 3$, then

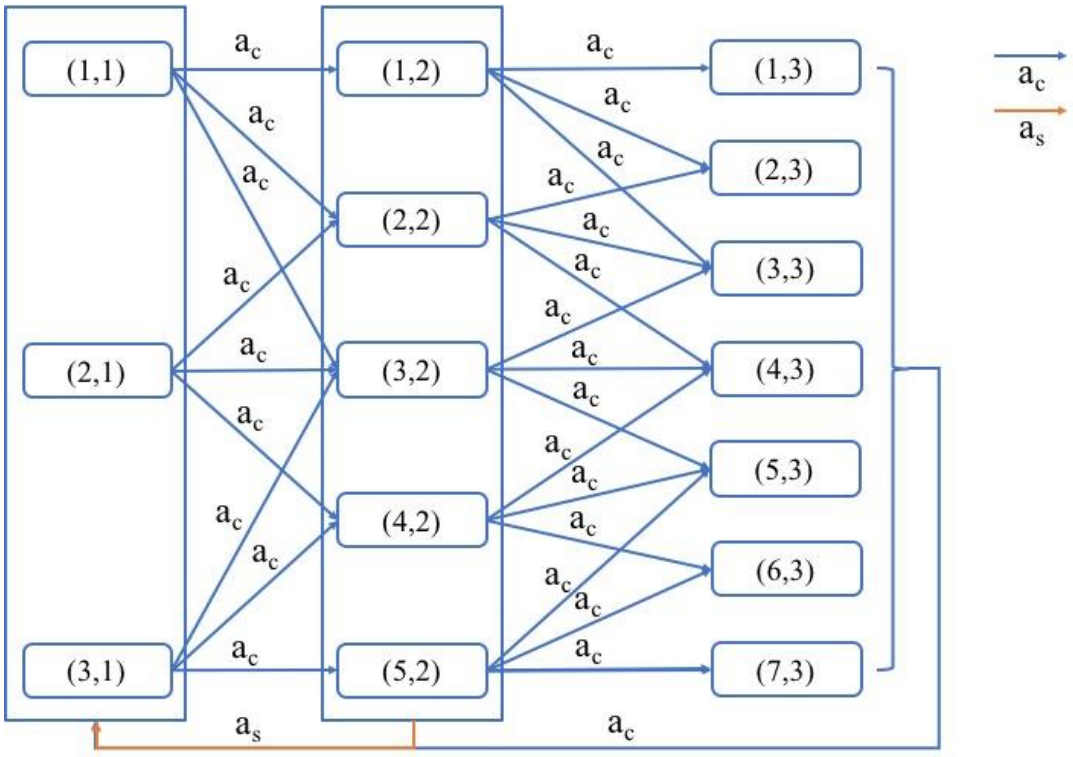


Figure 3.4 The schematic for a prototype of Markov Decision Process for this study. Numbers in each bracket represent the states. The first number in percentage represents the defective degree and the second integer number represents the stage or the observing. The stage is a key parameter to denote the completeness of the process.

$$S = \{(1,1),(2,1),(3,1),(1,2),(2,2),(3,2),(4,2),$$

$$(5,2),(1,3),(2,3),(3,3),(4,3),(5,3),(6,3),(7,3)\}.$$

And we suppose that

$$d_{1,1} = 0\%, d_{2,1} = 4.5\%, d_{3,1} = 9\%,$$

$$p_1 = 0.6, p_2 = 0.3, p_3 = 0.1.$$

Then $d_{1,2} = 0\%$, $d_{2,2} = 2.25\%$, $d_{3,2} = 4.5\%$, $d_{4,2} = 6.75\%$, $d_{5,2} = 9\%$, $d_{1,3} = 0\%$, $d_{2,3} = 1.5\%$, $d_{3,3} = 3\%$, $d_{4,3} = 4.5\%$, $d_{5,3} = 6\%$, $d_{6,3} = 7.5\%$, $d_{7,3} = 9\%$ and the actions are $A = \{a_s, a_c\} = \{0, 1\}$. From Figure 3.5, it is illustrated that if the process is completed, all states at the end will continue going back to the first stage $(1,1), (2,1)$ or $(3,1)$; if the process is stopped in the middle, the states will also go back to the first stage. The representations for states and actions are arbitrary and do not affect the actual MDP calculations. The transition matrix consists of two sub-matrices

$$P = [P(a_c), P(a_s)],$$

where the transition matrices for the two actions are simply approximated by

$$P_{(j,t),(j',t')}(a_c) = \begin{bmatrix} 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

and

$$
P_{(j,t),(j',t')}(a_s) = \begin{bmatrix}
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

we assume the transition probabilities between two states only depends on the transitioned state. The probability meets the sum rule, i.e., the probabilities of all possible states equals to 1. The transition probabilities for "stop" action is very simple, since no transition to other states is possible.

If we assume τ=20, $p_0 = 1$, $\psi = 0$ and $\gamma = 1.5$, so the finally completed part is valued as 1.5, then the corresponding reward matrix $R = [R(a_c), R(a_s)]$ can be hypothesized as

$$
R(a_c) = R(a_s) = [-0.333, -0.333, -0.333, -0.333, -0.333, -0.333, -0.333, -0.333,
1.167, 0.867, 0.567, 0.267, -0.033, -0.333, -0.733\ ]
$$

In writing this reward matrix, we mainly consider the following fact: (i) Assuming the cost of a whole AM part is unit 1, and on the first two stages the reward will be just the material cost of that stage, which will be $-1 \times \frac{1}{3} = -0.333$; (ii) a successfully produced object must have a value larger than 1, i.e., the value is increased. Here we take 1.5 units for a whole printed part; (iii) higher defect levels at stage 3 lead to the decreased value of the AM part; (iv) very high defect level at each stage results in terminating of the process and the part will be dumped, since it is not usable,

then a new building process will start. In this simple three step case, stopping at the intermediate step results in a loss of only the material cost.

3.3.2 Solution to the Numerical Example

Solution. With these assumptions and hypotheses, the MDP problem is displayed in detail and solved with the help of python code for Monte-Carlo simulation written by us. The optimal policy achieved from it is

$$\Pi^* = [a_c, a_c, a_s, a_c, a_c, a_c, a_c, a_s, a_c, a_c, a_c, a_c, a_c, a_c, a_c],$$

which means

$$\pi(1,1) = \pi(2,1) = \pi(1,2) = \pi(2,2) = \pi(3,2) = \pi(4,2) = \pi(1,3) = \pi(2,3) = \pi(3,3)$$
$$= \pi(4,3) = \pi(5,3) = \pi(6,3) = \pi(7,3) = a_c,$$
$$\pi(3,1) = \pi(5,2) = a_s.$$

In another word, except for state (3, 1) and (5, 2), the process should always continue, which is quite reasonable, since the defective level 9% is quite high already. The max total value at last is $V_{tot} = 0.0993$, a positive number. Monte-Carlo basic algorithms was used here to solve the problem and to show its convergence, the following plot is made.

If we adjust $\gamma = 2$ and keep τ=20, $p_0 = 1$, $\psi = 0$, then the corresponding reward matrix $R = [R(a_c), R(a_s)]$ can be hypothesized as

$$R(a_c) = R(a_s) = [-0.333, -0.333, -0.333, -0.333, -0.333, -0.333, -0.333, -0.333,$$
$$1.667, 1.367, 1.067, 0.767, 0.467, 0.167, -0.233]$$

The optimal policy will change to

$$\Pi^* = [a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c, a_c],$$

And the maximum total value at last will change to $V_{tot} = 0.5959$, which is also reasonable, since the gained value of completing a part will encourage the process to continue always.

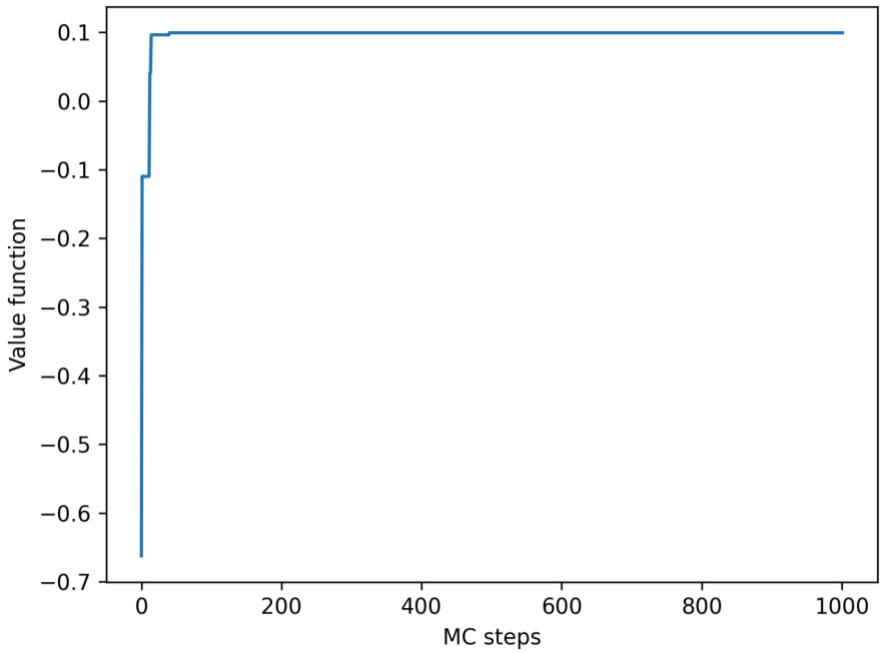


Figure 3.5 Total value variation with Monte-Carlo steps. It can clearly show that it is convergent around 100 steps.

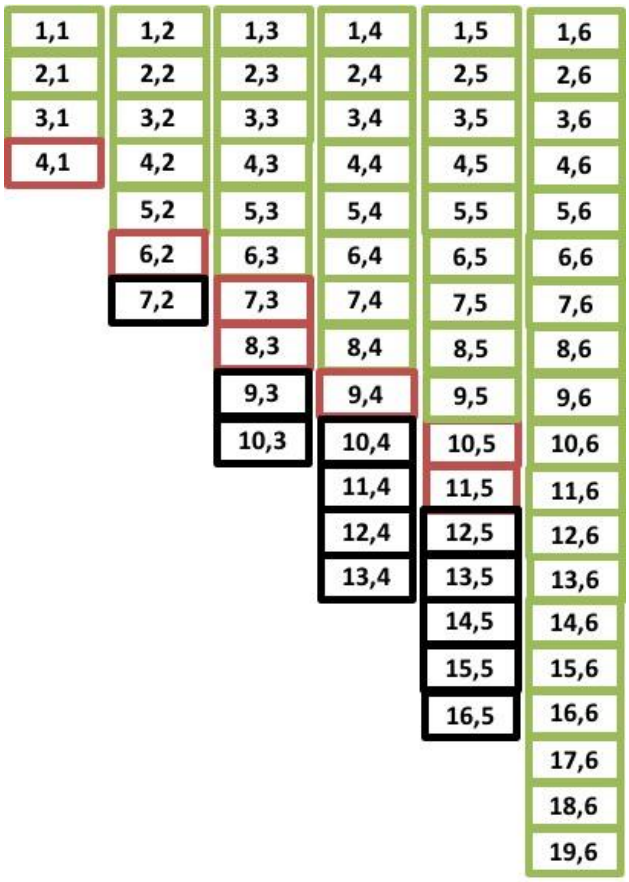


Figure 3.6 Optimal policies of one example obtained with value iteration

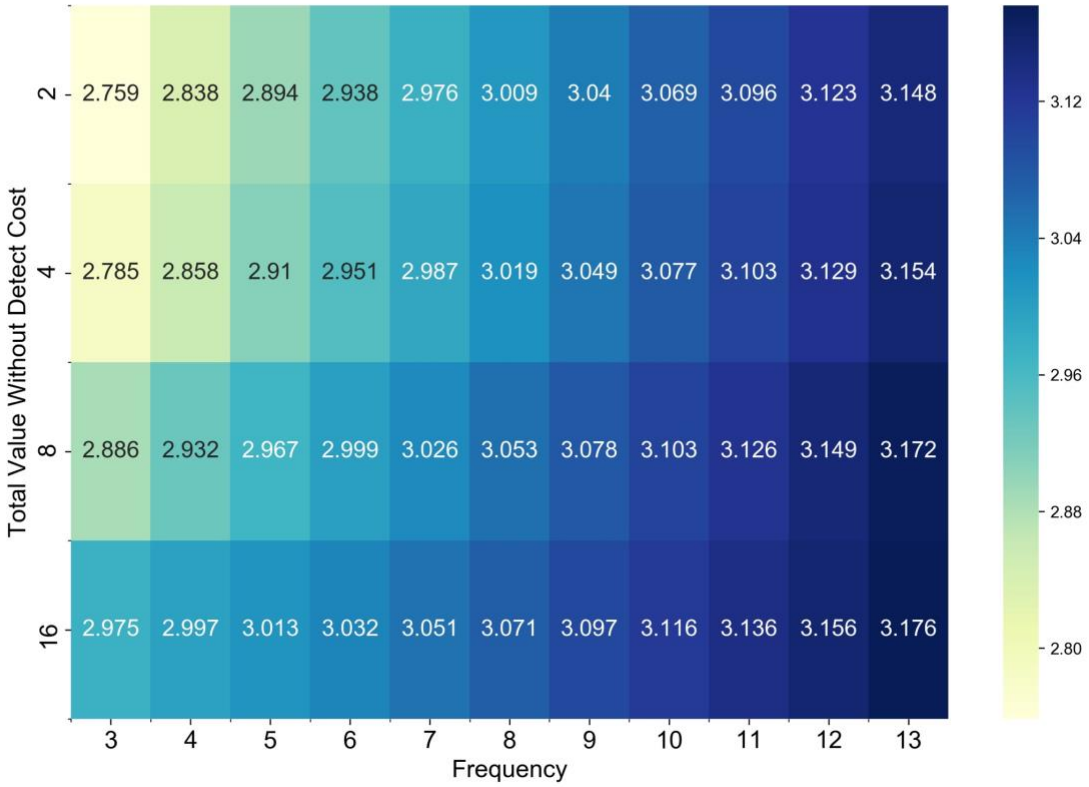


Figure 3.7 Heatmap of total values of a single part with various parameters without considering detecting cost

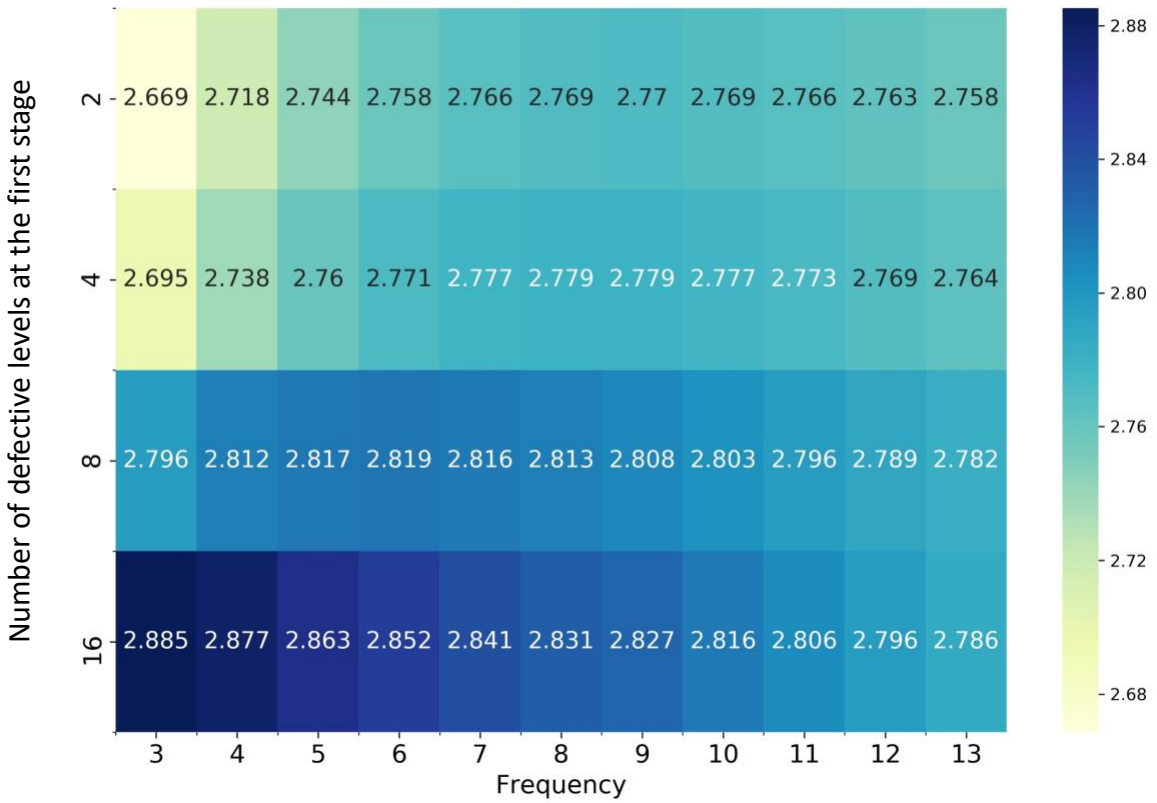


Figure 3.8 Heatmap of total values of a single part with various parameters considering detecting cost

This is just a simplified example, but it can briefly show the feasibility of our plan.

3.4 Discussion

When we increase the number of defective levels at first stage to four and assume there are six stages in the model, an optimal policy will be obtained through value iteration with the similar rewards and transition matrix used in section 3.3.1. A figure is plotted about the optimal policy on each state shown as in Figure 3.7. The green states mean policy of continue should be applied, while red and black states represent policy of stop should be conducted at those states. Black states are the states choose stop policy because of stop policy transferring from the last stage and red ones are those decided to be stopped at current stage.

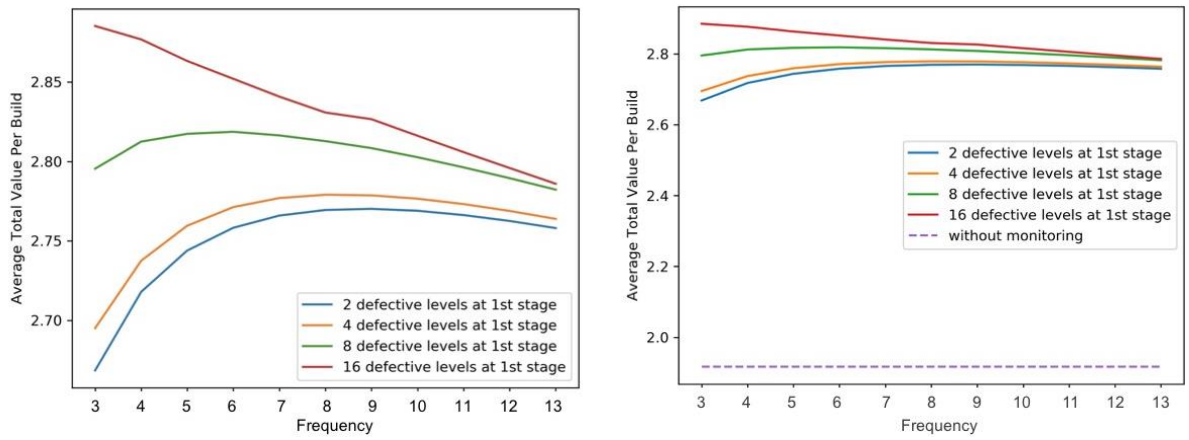
From the Figure 3.6, it is seen that there seems a clear boundary between the states choose to continue or stop actions. In each stage, if the policy is to stop at one specific state, then the policy of states that have higher defective levels will also choose stop policy. Applying this rule to solving process may also help to save time.

To find out the trend of total values of models with different detection frequencies f and number of defective levels at the first stage x, we assumed a whole set of parameters detection frequencies $F = [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]$ and the number of defective levels at first stage $X = [2, 4, 8, 16, 32]$. The reason we chose $X = [2, 4, 8, 16, 32]$ is that double the number of defective levels may help to remain the distribution of probabilities at each state. With the previous parameter set *F* and *X*, multiple models are built and solved with value iteration.

After the model is solved, the optimal policy a^* can be substituted into the MDP and the probabilities of each state P_{show} showing up can be calculated. Then the average value per stage can be achieved as $V_{avg} = P_{show} \cdot R(a^*)$ and the total value as $V_{tot\_wto\_detect\_cost} = V_{avg} \cdot T \; where \; T = F$. The total values of each model are recorded and plotted in the Figure 3.7. In this

figure, the detection cost is not considered. When we keep the number of defective levels at the first stage the same and change detection frequency from 3 to 13, then the total values increase; when we look at models with the same detection frequency but an increasing number of defective levels at the first stage, the values of them are raising as well. It is quite intuitive that when the detection frequency becomes higher, there are more detection stages for each part. When there are more defective levels at each stage, the defective conditions are refined. Both situations will help the system to detect the occurrence of defects earlier and as a result, it can terminate earlier to save cost and the values will increase accordingly. However, having more defective levels and higher detecting frequency significantly increase the state space of the MDP and cost huge computational burden.

In practice, it will cost time and money for detection. If the detection is frequent, quite an amount of cost will be added to the build of each AM part. The total value with detection cost subtracted will be $V_{tot} = V_{tot\_wto\_detect\_cost} - C_{detect} \cdot T$. These values of each model are calculated and represented in Figure 3.8. The figure shows that this time when we keep the number of defective levels at the first stage the same and change detection frequency from 3 to 13, then the total values usually increase and then decrease. The reason of the increasing of the values is explained above. The reason of the values decreasing after increasing is that if there are too many stages, the detection cost will increase substantially and thus the total values will be dragged lower. For this case, if the view is changed to another axis, it can be concluded that when the number of defective levels at first stage is rising, the total values are also increasing. It can be understood that if there are more defective levels at each stage, the model can find out a more accurate threshold to split the states with different policies, this should help in a way to make the total values raise. At the same time, the help will gradually fade when there are too many defective levels. Figure 3.8 also



(a) (b)

Figure 3.9 Trend of total values of a single part with various detection frequency and defective levels (a) and comparison with total value without monitor system (b).

shows that adding more defective levels may benefit the total cost more than increasing detecting frequency, but this observation strongly related to the detecting cost. When the detecting cost is low, increase frequency may be more beneficial. However, the computational time has been around 3 hours when the number of defective levels in stage 1 is 16 and the frequency is 13. Even though the MDP can be solved off-line, having much higher resolution seems computationally intractable.

To show the changing trends of total values with detection cost more obvious and compare them with the total value without a monitor system, Figure 3.9 is plotted. Figure 3.9 (a) directly plots the total values of different models in lines and Figure 3.9 (b) adds a dotted line at the bottom which represents the total value without any monitor system. The average value of the all the total values with our monitor system is around 2.79 and the total value without any monitor system is 1.91. Then finally the effectiveness of the proposed ML-MDP framework for improving AM productivity by (2.79-1.91)/1.91 = 46%.

3.5 Conclusion

With our proposed MDP model, combining with the previous ML models, a closed-loop quality control system is built up for AM. Value iteration is applied most of time in this chapter. The optimal policy of our MDP models has some specific properties that may be further utilized in solving process to save computational time. For example, at one stage, when the optimal action is "stop" for a defective level, the optimal action for a worse defective level should also be "stop". When we consider the number of defective levels and frequency of detection in our ML-MDP framework, some patterns emerge on the total values of these models. In a nutshell, dividing more defective levels in each stage and more frequent detection during AM process will help to improve total values of the AM process itself, but when it exceeds some extent, the total values may not be

improved much and even be reduced because of the cost associated with detecting effort and the high computational burden caused by the large state space. When the state space is too large, the value iteration algorithm may not yield a quality solution in a reasonable amount of computational time. The results clearly show that the proposed ML-MDP monitor process can significantly improve the productivity of the AM process, compared against the process without any monitoring mechanism. Although there are still many obstacles and our idea has not been applied in real production, it offers a way or a thought to solve the challenges in real-time monitoring of AM process.

In the future, experimental studies are necessary to validate the proposed ML-MDP framework. The optimality properties should be explored and incorporated into the value iteration or policy iteration algorithm to improve the computational efficiency to solve the MDP so that more defective levels and higher detecting frequency could be considered to further improve the AM productivity. In this research, we assume the underlying machine condition is stationary. However, AM machines often face various deterioration, such as higher oxygen level or inaccurate laser intensity. The ML-MDP framework may be used to detect unhealthy machine conditions and introduce more actions, such as machine maintenance or adjusting machine settings in real time (e.g., adding more nitrogen). It may also be used to detect problems caused by feedstock and designs. However, all of them require much more sophisticated ML and MDP models to separate different reasons that may cause AM quality problems and it is expected that the computational burden will be a challenge in practice.

References

3D Barrel. (n.d.). Retrieved from https://www.turbosquid.com/3d-models/old-barrel-3d-model-1217355

Abraham, A., & Elrahman, S. M. A. (2013). A Review of Class Imbalance Problem. *Journal of Network and Innovative Computing*, *1*, 332–340. Retrieved from www.mirlabs.net/jnic/index.html

Alpaydın, E. (2009). *An Introduction to Machine Learning*. (T. Dietterich, C. Bishop, D. Heckerman, M. Jordan, & M. Kearns, Eds.), *Clinical Pharmacology and Therapeutics* (second). London, England: The MIT Press Cambridge,. https://doi.org/10.1002/cpt.1796

Altman, N. S. (1991). An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician*, *46*(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

Aluze, D., Merienne, F., Dumont, C., & Gorria, P. (2002). Vision system for defect imaging, detection, and characterization on a specular surface of a 3D object. *Image and Vision Computing*, *20*(8), 569–580. https://doi.org/10.1016/S0262-8856(02)00046-X

Amit, Y., & Geman, D. (1997). Shape Quantization and Recognition with Randomized Trees. *Neural Computation*, *9*(7), 1545–1588. https://doi.org/10.1162/neco.1997.9.7.1545

Arian, R., Hariri, A., Mehridehnavi, A., Fassihi, A., & Ghasemi, F. (2020). Protein kinase inhibitors' classification using K-Nearest neighbor algorithm. *Computational Biology and Chemistry*, *86*, 107269. https://doi.org/10.1016/j.compbiolchem.2020.107269

Attaran, M. (2017). The rise of 3-D printing: The advantages of additive manufacturing over traditional manufacturing. *Business Horizons*, *60*(5), 677–688.

https://doi.org/10.1016/j.bushor.2017.05.011

Barua, S., Liou, F., Newkirk, J., & Sparks, T. (2014). Vision-based defect detection in laser metal deposition process. *Rapid Prototyping Journal*, *20*(1), 77–86. https://doi.org/10.1108/RPJ-04-2012-0036

Baxter, L. A., & Puterman, M. L. (1995). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. *Technometrics* (Vol. 37). https://doi.org/10.2307/1269932

Benda, J. A. (1994). Temperature-Controlled Selective Laser Sintering. *DTICDocument*, 277–284. Retrieved from https://repositories.lib.utexas.edu/handle/2152/68655

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, *24*(2), 123–140. https://doi.org/10.1023/A:1018054314350

Breiman, L. (1997). *Arcing the edge*. *Statistics* (Vol. 4).

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Campbell, I., Bourell, D., & Gibson, I. (2012). Additive manufacturing : rapid prototyping comes of age. *Rapid Prototyping Journal*, *18*, 255–258. https://doi.org/10.1108/13552541211231563

Campbell, R. I., Martorelli, M., & Lee, H. S. (2002). Surface roughness visualisation for rapid prototyping models. *CAD Computer Aided Design*, *34*(10), 717–725. https://doi.org/10.1016/S0010-4485(01)00201-9

Chadès, I., Chapron, G., Cros, M. J., Garcia, F., & Sabbadin, R. (2014). MDPtoolbox: A multi-platform toolbox to solve stochastic dynamic programming problems. *Ecography*, *37*(9), 916–920. https://doi.org/10.1111/ecog.00888

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Cohen, A., Laviv, A., Berman, P., Nashef, R., & Abu-Tair, J. (2009). Mandibular reconstruction using stereolithographic 3-dimensional printing modeling technology. *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology and Endodontology*, *108*(5), 661–666. https://doi.org/10.1016/j.tripleo.2009.05.023

Cortes, C., & Vladimir, V. (1995). Support-Vector Networks. *Machine Learning*, *20*, 273–297.

Dastoorian, R., Elhabashy, A. E., Tian, W., Wells, L. J., & Camelio, J. A. (2018). Automated Surface Inspection Using 3D Point Cloud Data in Manufacturing: A Case Study. *Volume 3: Manufacturing Equipment and Systems*, (September), V003T02A036. https://doi.org/10.1115/MSEC2018-6542

Dawson-Haggerty, M. (2019). Trimesh [Computer software]. Retrieved from https://trimsh.org/

Delli, U., & Chang, S. (2018). Automated Process Monitoring in 3D Printing Using Supervised Machine Learning. *Procedia Manufacturing*, *26*, 865–870. https://doi.org/10.1016/j.promfg.2018.07.111

Douglas, S., & Stanley, W. (2015). Costs and Cost Effectiveness of Additive Manufacturing A Literature Review and Discussion. *NIST Special Publication 1176*, (October 2016).

"EinScan-SP." (n.d.). Retrieved from https://www.einscan.com/desktop-3d-scanners/einscan-sp/

Everton, S. K., Hirsch, M., Stavroulakis, P. I., Leach, R. K., & Clare, A. T. (2016). Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. *Materials and Design*, *95*, 431–445. https://doi.org/10.1016/j.matdes.2016.01.099

Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., & Lin, C. J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, *9*, 1871–1874. https://doi.org/10.1145/1390681.1442794

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139. https://doi.org/10.1006/JCSS.1997.1504

Friedman, J. H. (2001). Greedy Function Approximation : A Gradient Boosting Machine 1 Function estimation 2 Numerical optimization in function space. *The Annuals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.2307/2699986

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. https://doi.org/10.1109/TSMCC.2011.2161285

Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, *2*(4), 42–47. Retrieved from http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf

Gao, W., Zhang, Y., Ramanujan, D., Ramani, K., Chen, Y., Williams, C. B., … Zavattieri, P. D. (2015). The status, challenges, and future of additive manufacturing in engineering. *CAD Computer Aided Design*, *69*, 65–89. https://doi.org/10.1016/j.cad.2015.04.001

Gibson, I., Rosen, D., & Stucker, B. (2015). *Additive manufacturing technologies: 3D printing, rapid prototyping, and direct digital manufacturing, second edition*. *Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing, Second Edition*. Springer New York. https://doi.org/10.1007/978-1-4939-2113-3

Girardeau-Montaut, D., Maloney, A., & Janvier, R. (2019). CloudCompare [GPL software]. Retrieved from https://github.com/CloudCompare/CloudCompare/releases/tag/v2.10.3

Gobert, C., Reutzel, E. W., Petrich, J., Nassar, A. R., & Phoha, S. (2018). Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging. *Additive Manufacturing*, *21*, 517–528. https://doi.org/10.1016/j.addma.2018.04.005

Golovinskiy, A., & Funkhouser, T. (2009). Consistent segmentation of 3D models. *Computers and Graphics (Pergamon)*, *33*(3), 262–269. https://doi.org/10.1016/j.cag.2009.03.010

Guo, N., & Leu, M. C. (2013, September 8). Additive manufacturing: Technology, applications and research needs. *Frontiers of Mechanical Engineering*. Springer. https://doi.org/10.1007/s11465-013-0248-8

Ha, C. W., & Yang, D. Y. (2014). Fabrication of micro open structure using 3D laser scanning method in nano-stereolithography. In *2014 International Conference on Manipulation, Manufacturing and Measurement on the Nanoscale, 3M-NANO 2014 - Conference Proceedings* (pp. 299–303). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/3M-NANO.2014.7057311

Harding, J. A., Shahbaz, M., Srinivas, & Kusiak, A. (2006). Data Mining in Manufacturing: A Review. *Journal of Manufacturing Science and Engineering*, *128*(4), 969–976. https://doi.org/10.1115/1.2194554

Holzmond, O., & Li, X. (2017). In situ real time defect detection of 3D printed parts. *Additive Manufacturing*, *17*, 135–142. https://doi.org/10.1016/j.addma.2017.08.003

Huang, D., Du, S., Li, G., Zhao, C., & Deng, Y. (2018). Detection and monitoring of defects on

three-dimensional curved surfaces based on high-density point cloud data. *Precision Engineering*, *53*, 79–95. https://doi.org/10.1016/J.PRECISIONENG.2018.03.001

Huang, S. H., Liu, P., Mokasdar, A., & Hou, L. (2013, July 16). Additive manufacturing and its societal impact: A literature review. *International Journal of Advanced Manufacturing Technology*. Springer. https://doi.org/10.1007/s00170-012-4558-5

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, *6*(5), 429–449. https://doi.org/10.3233/ida-2002-6504

Kalogerakis, E., Hertzmann, A., & Singh, K. (2010). Learning 3D mesh segmentation and labeling. In *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10* (p. 1). New York, New York, USA: Association for Computing Machinery (ACM). https://doi.org/10.1145/1833349.1778839

Kim, H., Lin, Y., & Tseng, T. L. B. (2018). A review on quality control in additive manufacturing. *Rapid Prototyping Journal*, *24*(3), 645–669. https://doi.org/10.1108/RPJ-03-2017-0048

Kwak, D.-S., & Kim, K.-J. (2012). A data mining approach considering missing values for the optimization of semiconductor-manufacturing processes. *Expert Systems with Applications*, *39*(3), 2590–2596. https://doi.org/10.1016/J.ESWA.2011.08.114

Leach, R. (2011). *Optical Measurement of Surface Topography*. (P. R. Leach, Ed.). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12012-1

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). *Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning*. *Journal of Machine Learning Research* (Vol. 18). Retrieved from http://jmlr.org/papers/v18/16-365.html.

Lin, W., Shen, H., Fu, J., & Wu, S. (2019). Online Quality Monitoring in Material Extrusion Additive Manufacturing Processes based on Laser Scanning Technology. *Precision*

Engineering. https://doi.org/10.1016/j.precisioneng.2019.06.004

Liu, R., & Zhang, H. (2004). Segmentation of 3D meshes through spectral clustering. In *Proceedings - Pacific Conference on Computer Graphics and Applications* (pp. 298–305). https://doi.org/10.1109/PCCGA.2004.1348360

Lott, P., Schleifenbaum, H., Meiners, W., Wissenbach, K., Hinke, C., & Bültmann, J. (2011). Design of an optical system for the in situ process monitoring of Selective Laser Melting (SLM). In *Physics Procedia* (Vol. 12, pp. 683–690). Elsevier B.V. https://doi.org/10.1016/j.phpro.2011.03.085

Lu, Q. Y., & Wong, C. H. (2018). Additive manufacturing process monitoring and control by non-destructive testing techniques: challenges and in-process monitoring. *Virtual and Physical Prototyping*, *13*(2), 39–48. https://doi.org/10.1080/17452759.2017.1351201

Machado, B., Rodrigues, T., Lopes, Z., Lopes, R., & Mesquita, M. (2013). Paraparesis: A rare presentation of thrombosis of the abdominal aorta. *European Journal of Internal Medicine*, *24*(1), e256. https://doi.org/10.1016/j.ejim.2013.08.659

Madrigal, C. A., Branch, J. W., Restrepo, A., & Mery, D. (2017). A method for automatic surface inspection using a model-based 3D descriptor. *Sensors (Switzerland)*, *17*(10), 1–20. https://doi.org/10.3390/s17102262

Mandache, C. (2019). Overview of non-destructive evaluation techniques for metal-based additive manufacturing. *Materials Science and Technology (United Kingdom)*, *35*(9), 1007–1015. https://doi.org/10.1080/02670836.2019.1596370

Moreo, A., Esuli, A., & Sebastiani, F. (2016). Distributional random oversampling for imbalanced text classification. In *SIGIR 2016 - Proceedings of the 39th International ACM SIGIR*

Conference on Research and Development in Information Retrieval (pp. 805–808). New York, New York, USA: Association for Computing Machinery, Inc. https://doi.org/10.1145/2911451.2914722

Nayar, S. K., Sanderson, A. C., Weiss, E., & Simon, D. A. (1990). Specular Surface Inspection Using Structured Highlight and Gaussian Images. *IEEE Transactions on Robotics and Automation*. https://doi.org/10.1109/70.54736

Opitz, D., & Maclin, R. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, *11*, 169–198. https://doi.org/10.1613/jair.614

Pedregosa, F., Varoquaux, G., Buitinck, L., Louppe, G., Grisel, O., Pedregosa, F., & Mueller, A. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(1), 29–33. https://doi.org/10.1145/2786984.2786995

Perard, D., & Beyerer, J. (1997). Three-dimensional measurement of specular free-form surfaces with a structured-lighting reflection technique. In K. G. Harding & D. J. Svetkoff (Eds.), *Three-Dimensional Imaging and Laser-based Systems for Metrology and Inspection III* (Vol. 3204, pp. 74–80). https://doi.org/10.1117/12.294443

Pham, D. T., & Afify, A. A. (2005). Machine-learning techniques and their applications in manufacturing. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *219*(5), 395–412. https://doi.org/10.1243/095440505X32274

Puterman, M. L., & Shin, M. C. (1978). Modified Policy Iteration Algorithms for Discounted Markov Decision Problems. *Management Science*, *24*(11), 1127–1137. https://doi.org/10.1287/mnsc.24.11.1127

Sarghie, B., Costea, M., & Liute, D. (2013). *Anthropometric study of the foot using 3d scanning*

method and statistical analysis. *International Symposium in Knitting and Apparel* (Vol. June). Retrieved from https://www.researchgate.net/publication/242019927

Satija, H., Amortila, P., & Pineau, J. (2020). *Constrained Markov Decision Processes via Backward Value Functions*.

Scime, L., & Beuth, J. (2019). Using machine learning to identify in-situ melt pool signatures indicative of flaw formation in a laser powder bed fusion additive manufacturing process. *Additive Manufacturing*, *25*, 151–165. https://doi.org/10.1016/j.addma.2018.11.010

Seifi, M., Salem, A., Beuth, J., Harrysson, O., & Lewandowski, J. J. (2016). Overview of Materials Qualification Needs for Metal Additive Manufacturing. *JOM*, *68*(3), 747–764. https://doi.org/10.1007/s11837-015-1810-0

Shen, H., Li, S., Gu, D., & Chang, H. (2012). Bearing defect inspection based on machine vision. *Measurement*, *45*, 719–733. https://doi.org/10.1016/j.measurement.2011.12.018

Sitthi-Amorn, P., Ramos, J. E., Wang, Y., Kwan, J., Lan, J., Wang, W., & Matusik, W. (2015). MultiFab: A machine vision assisted platform for multi-material 3D printing. *ACM Transactions on Graphics*, *34*(4). https://doi.org/10.1145/2766962

Sood, A. K., Ohdar, R. K., & Mahapatra, S. S. (2009). Improving dimensional accuracy of Fused Deposition Modelling processed part using grey Taguchi method. *Materials and Design*, *30*(10), 4243–4252. https://doi.org/10.1016/j.matdes.2009.04.030

Suykens, J. A. K., & Vandewalle, J. (1999). Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, *9*(3), 293–300. https://doi.org/10.1023/A:1018628609742

Tang, P., Akinci, B., & Huber, D. (2008). Characterization of three algorithms for detecting surface flatness defects from dense point clouds. In J. A. Beraldin, G. S. Cheok, M. McCarthy, & U.

Neuschaefer-Rube (Eds.), *Three-Dimensional Imaging Metrology* (Vol. 7239, p. 72390N). International Society for Optics and Photonics. https://doi.org/10.1117/12.805727

Tharwat, A. (2018). Classification assessment methods. *Applied Computing and Informatics*. https://doi.org/10.1016/J.ACI.2018.08.003

Watkins, C., & Holloway, R. (1992). *Technical Note : Q-Learning* (Vol. 8).

Westerweel, B., Basten, R., den Boer, J., & van Houtum, G. (2020). *Printing Spare Parts at Remote Locations: Fulfilling the Promise of Additive Manufacturing*. *Production and Operations Management*. https://doi.org/10.1111/poms.13298

Wuest, T., Weimer, D., Irgens, C., & Thoben, K. D. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. *Production and Manufacturing Research*, *4*(1), 23–45. https://doi.org/10.1080/21693277.2016.1192517

Yadroitsev, I., Krakhmalev, P., & Yadroitsava, I. (2014). Selective laser melting of Ti6Al4V alloy for biomedical applications: Temperature monitoring and microstructural evolution. *Journal of Alloys and Compounds*, *583*, 404–409. https://doi.org/10.1016/j.jallcom.2013.08.183

Yao, B., Imani, F., & Yang, H. (2018). Markov decision process for image-guided additive manufacturing. *IEEE Robotics and Automation Letters*, *3*(4), 2792–2798. https://doi.org/10.1109/LRA.2018.2839973

Yao, B., & Yang, H. (2018). Constrained markov decision process modeling for sequential optimization of additive manufacturing build quality. *IEEE Access*, *6*, 54783–54794. https://doi.org/10.1109/ACCESS.2018.2872391

Yong, Y. (2012). The Research of Imbalanced Data Set of Sample Sampling Method Based on K-Means Cluster and Genetic Algorithm. *Energy Procedia*, *17*, 164–170.

https://doi.org/10.1016/J.EGYPRO.2012.02.078

Zeng, K., Pal, D., & Stucker, B. (2012). A review of thermal analysis methods in laser sintering and selective laser melting. *23rd Annual International Solid Freeform Fabrication Symposium - An Additive Manufacturing Conference, SFF 2012*, 796–814.

Zhang, N. L., & Zhang, W. (2001). Speeding up the convergence of value iteration in partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, *14*, 29–51. https://doi.org/10.1613/jair.761

Zhihua, Z. (2012). Book Review: Ensemble Methods: Foundations and Algorithms. *IEEE Intelligent Informatics Bulletin*, *13*.

Zhou, Q.-Y., Park, J., & Koltun, V. (2018, January 29). Open3D: A Modern Library for 3D Data Processing [Computer Software]. Retrieved from http://arxiv.org/abs/1801.09847

3D Barrel. (n.d.). Retrieved from https://www.turbosquid.com/3d-models/old-barrel-3d-model-1217355

Abraham, A., & Elrahman, S. M. A. (2013). A Review of Class Imbalance Problem. *Journal of Network and Innovative Computing*, *1*, 332–340. Retrieved from www.mirlabs.net/jnic/index.html

Alpaydın, E. (2009). *An Introduction to Machine Learning*. (T. Dietterich, C. Bishop, D. Heckerman, M. Jordan, & M. Kearns, Eds.), *Clinical Pharmacology and Therapeutics* (second). London, England: The MIT Press Cambridge,. https://doi.org/10.1002/cpt.1796

Altman, N. S. (1991). An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician*, *46*(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

Aluze, D., Merienne, F., Dumont, C., & Gorria, P. (2002). Vision system for defect imaging,

detection, and characterization on a specular surface of a 3D object. *Image and Vision Computing*, *20*(8), 569–580. https://doi.org/10.1016/S0262-8856(02)00046-X

Amit, Y., & Geman, D. (1997). Shape Quantization and Recognition with Randomized Trees. *Neural Computation*, *9*(7), 1545–1588. https://doi.org/10.1162/neco.1997.9.7.1545

Arian, R., Hariri, A., Mehridehnavi, A., Fassihi, A., & Ghasemi, F. (2020). Protein kinase inhibitors' classification using K-Nearest neighbor algorithm. *Computational Biology and Chemistry*, *86*, 107269. https://doi.org/10.1016/j.compbiolchem.2020.107269

Attaran, M. (2017). The rise of 3-D printing: The advantages of additive manufacturing over traditional manufacturing. *Business Horizons*, *60*(5), 677–688. https://doi.org/10.1016/j.bushor.2017.05.011

Barua, S., Liou, F., Newkirk, J., & Sparks, T. (2014). Vision-based defect detection in laser metal deposition process. *Rapid Prototyping Journal*, *20*(1), 77–86. https://doi.org/10.1108/RPJ-04-2012-0036

Baxter, L. A., & Puterman, M. L. (1995). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. *Technometrics* (Vol. 37). https://doi.org/10.2307/1269932

Benda, J. A. (1994). Temperature-Controlled Selective Laser Sintering. *DTICDocument*, 277–284. Retrieved from https://repositories.lib.utexas.edu/handle/2152/68655

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, *24*(2), 123–140. https://doi.org/10.1023/A:1018054314350

Breiman, L. (1997). *Arcing the edge*. *Statistics* (Vol. 4).

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Campbell, I., Bourell, D., & Gibson, I. (2012). Additive manufacturing : rapid prototyping comes of age. *Rapid Prototyping Journal*, *18*, 255–258. https://doi.org/10.1108/13552541211231563

Campbell, R. I., Martorelli, M., & Lee, H. S. (2002). Surface roughness visualisation for rapid prototyping models. *CAD Computer Aided Design*, *34*(10), 717–725. https://doi.org/10.1016/S0010-4485(01)00201-9

Chadès, I., Chapron, G., Cros, M. J., Garcia, F., & Sabbadin, R. (2014). MDPtoolbox: A multi-platform toolbox to solve stochastic dynamic programming problems. *Ecography*, *37*(9), 916–920. https://doi.org/10.1111/ecog.00888

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Cohen, A., Laviv, A., Berman, P., Nashef, R., & Abu-Tair, J. (2009). Mandibular reconstruction using stereolithographic 3-dimensional printing modeling technology. *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology and Endodontology*, *108*(5), 661–666. https://doi.org/10.1016/j.tripleo.2009.05.023

Cortes, C., & Vladimir, V. (1995). Support-Vector Networks. *Machine Learning*, *20*, 273–297.

Dastoorian, R., Elhabashy, A. E., Tian, W., Wells, L. J., & Camelio, J. A. (2018). Automated Surface Inspection Using 3D Point Cloud Data in Manufacturing: A Case Study. *Volume 3: Manufacturing Equipment and Systems*, (September), V003T02A036. https://doi.org/10.1115/MSEC2018-6542

Dawson-Haggerty, M. (2019). Trimesh [Computer software]. Retrieved from https://trimsh.org/

Delli, U., & Chang, S. (2018). Automated Process Monitoring in 3D Printing Using Supervised Machine Learning. *Procedia Manufacturing*, *26*, 865–870. https://doi.org/10.1016/j.promfg.2018.07.111

Douglas, S., & Stanley, W. (2015). Costs and Cost Effectiveness of Additive Manufacturing A Literature Review and Discussion. *NIST Special Publication 1176*, (October 2016).

"EinScan-SP." (n.d.). Retrieved from https://www.einscan.com/desktop-3d-scanners/einscan-sp/

Everton, S. K., Hirsch, M., Stavroulakis, P. I., Leach, R. K., & Clare, A. T. (2016). Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. *Materials and Design*, *95*, 431–445. https://doi.org/10.1016/j.matdes.2016.01.099

Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., & Lin, C. J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, *9*, 1871–1874. https://doi.org/10.1145/1390681.1442794

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139. https://doi.org/10.1006/JCSS.1997.1504

Friedman, J. H. (2001). Greedy Function Approximation : A Gradient Boosting Machine 1 Function estimation 2 Numerical optimization in function space. *The Annuals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.2307/2699986

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. https://doi.org/10.1109/TSMCC.2011.2161285

Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets.

International Journal of Emerging Technology and Advanced Engineering, *2*(4), 42–47. Retrieved from http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf

Gao, W., Zhang, Y., Ramanujan, D., Ramani, K., Chen, Y., Williams, C. B., … Zavattieri, P. D. (2015). The status, challenges, and future of additive manufacturing in engineering. *CAD Computer Aided Design*, *69*, 65–89. https://doi.org/10.1016/j.cad.2015.04.001

Gibson, I., Rosen, D., & Stucker, B. (2015). *Additive manufacturing technologies: 3D printing, rapid prototyping, and direct digital manufacturing, second edition*. *Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing, Second Edition*. Springer New York. https://doi.org/10.1007/978-1-4939-2113-3

Girardeau-Montaut, D., Maloney, A., & Janvier, R. (2019). CloudCompare [GPL software]. Retrieved from https://github.com/CloudCompare/CloudCompare/releases/tag/v2.10.3

Gobert, C., Reutzel, E. W., Petrich, J., Nassar, A. R., & Phoha, S. (2018). Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging. *Additive Manufacturing*, *21*, 517–528. https://doi.org/10.1016/j.addma.2018.04.005

Golovinskiy, A., & Funkhouser, T. (2009). Consistent segmentation of 3D models. *Computers and Graphics (Pergamon)*, *33*(3), 262–269. https://doi.org/10.1016/j.cag.2009.03.010

Guo, N., & Leu, M. C. (2013, September 8). Additive manufacturing: Technology, applications and research needs. *Frontiers of Mechanical Engineering*. Springer. https://doi.org/10.1007/s11465-013-0248-8

Ha, C. W., & Yang, D. Y. (2014). Fabrication of micro open structure using 3D laser scanning method in nano-stereolithography. In *2014 International Conference on Manipulation,*

Manufacturing and Measurement on the Nanoscale, 3M-NANO 2014 - Conference Proceedings (pp. 299–303). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/3M-NANO.2014.7057311

Harding, J. A., Shahbaz, M., Srinivas, & Kusiak, A. (2006). Data Mining in Manufacturing: A Review. *Journal of Manufacturing Science and Engineering*, *128*(4), 969–976. https://doi.org/10.1115/1.2194554

Holzmond, O., & Li, X. (2017). In situ real time defect detection of 3D printed parts. *Additive Manufacturing*, *17*, 135–142. https://doi.org/10.1016/j.addma.2017.08.003

Huang, D., Du, S., Li, G., Zhao, C., & Deng, Y. (2018). Detection and monitoring of defects on three-dimensional curved surfaces based on high-density point cloud data. *Precision Engineering*, *53*, 79–95. https://doi.org/10.1016/J.PRECISIONENG.2018.03.001

Huang, S. H., Liu, P., Mokasdar, A., & Hou, L. (2013, July 16). Additive manufacturing and its societal impact: A literature review. *International Journal of Advanced Manufacturing Technology*. Springer. https://doi.org/10.1007/s00170-012-4558-5

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, *6*(5), 429–449. https://doi.org/10.3233/ida-2002-6504

Kalogerakis, E., Hertzmann, A., & Singh, K. (2010). Learning 3D mesh segmentation and labeling. In *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10* (p. 1). New York, New York, USA: Association for Computing Machinery (ACM). https://doi.org/10.1145/1833349.1778839

Kim, H., Lin, Y., & Tseng, T. L. B. (2018). A review on quality control in additive manufacturing. *Rapid Prototyping Journal*, *24*(3), 645–669. https://doi.org/10.1108/RPJ-03-2017-0048

Kwak, D.-S., & Kim, K.-J. (2012). A data mining approach considering missing values for the

optimization of semiconductor-manufacturing processes. *Expert Systems with Applications*, *39*(3), 2590–2596. https://doi.org/10.1016/J.ESWA.2011.08.114

Leach, R. (2011). *Optical Measurement of Surface Topography*. (P. R. Leach, Ed.). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12012-1

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). *Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning*. *Journal of Machine Learning Research* (Vol. 18). Retrieved from http://jmlr.org/papers/v18/16-365.html.

Lin, W., Shen, H., Fu, J., & Wu, S. (2019). Online Quality Monitoring in Material Extrusion Additive Manufacturing Processes based on Laser Scanning Technology. *Precision Engineering*. https://doi.org/10.1016/j.precisioneng.2019.06.004

Liu, R., & Zhang, H. (2004). Segmentation of 3D meshes through spectral clustering. In *Proceedings - Pacific Conference on Computer Graphics and Applications* (pp. 298–305). https://doi.org/10.1109/PCCGA.2004.1348360

Lott, P., Schleifenbaum, H., Meiners, W., Wissenbach, K., Hinke, C., & Bültmann, J. (2011). Design of an optical system for the in situ process monitoring of Selective Laser Melting (SLM). In *Physics Procedia* (Vol. 12, pp. 683–690). Elsevier B.V. https://doi.org/10.1016/j.phpro.2011.03.085

Lu, Q. Y., & Wong, C. H. (2018). Additive manufacturing process monitoring and control by non-destructive testing techniques: challenges and in-process monitoring. *Virtual and Physical Prototyping*, *13*(2), 39–48. https://doi.org/10.1080/17452759.2017.1351201

Machado, B., Rodrigues, T., Lopes, Z., Lopes, R., & Mesquita, M. (2013). Paraparesis: A rare presentation of thrombosis of the abdominal aorta. *European Journal of Internal Medicine*,

24(1), e256. https://doi.org/10.1016/j.ejim.2013.08.659

Madrigal, C. A., Branch, J. W., Restrepo, A., & Mery, D. (2017). A method for automatic surface inspection using a model-based 3D descriptor. *Sensors (Switzerland)*, *17*(10), 1–20. https://doi.org/10.3390/s17102262

Mandache, C. (2019). Overview of non-destructive evaluation techniques for metal-based additive manufacturing. *Materials Science and Technology (United Kingdom)*, *35*(9), 1007–1015. https://doi.org/10.1080/02670836.2019.1596370

Moreo, A., Esuli, A., & Sebastiani, F. (2016). Distributional random oversampling for imbalanced text classification. In *SIGIR 2016 - Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 805–808). New York, New York, USA: Association for Computing Machinery, Inc. https://doi.org/10.1145/2911451.2914722

Nayar, S. K., Sanderson, A. C., Weiss, E., & Simon, D. A. (1990). Specular Surface Inspection Using Structured Highlight and Gaussian Images. *IEEE Transactions on Robotics and Automation*. https://doi.org/10.1109/70.54736

Opitz, D., & Maclin, R. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, *11*, 169–198. https://doi.org/10.1613/jair.614

Pedregosa, F., Varoquaux, G., Buitinck, L., Louppe, G., Grisel, O., Pedregosa, F., & Mueller, A. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(1), 29–33. https://doi.org/10.1145/2786984.2786995

Perard, D., & Beyerer, J. (1997). Three-dimensional measurement of specular free-form surfaces with a structured-lighting reflection technique. In K. G. Harding & D. J. Svetkoff (Eds.),

Three-Dimensional Imaging and Laser-based Systems for Metrology and Inspection III (Vol. 3204, pp. 74–80). https://doi.org/10.1117/12.294443

Pham, D. T., & Afify, A. A. (2005). Machine-learning techniques and their applications in manufacturing. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *219*(5), 395–412. https://doi.org/10.1243/095440505X32274

Puterman, M. L., & Shin, M. C. (1978). Modified Policy Iteration Algorithms for Discounted Markov Decision Problems. *Management Science*, *24*(11), 1127–1137. https://doi.org/10.1287/mnsc.24.11.1127

Sarghie, B., Costea, M., & Liute, D. (2013). *Anthropometric study of the foot using 3d scanning method and statistical analysis*. *International Symposium in Knitting and Apparel* (Vol. June). Retrieved from https://www.researchgate.net/publication/242019927

Satija, H., Amortila, P., & Pineau, J. (2020). *Constrained Markov Decision Processes via Backward Value Functions*.

Scime, L., & Beuth, J. (2019). Using machine learning to identify in-situ melt pool signatures indicative of flaw formation in a laser powder bed fusion additive manufacturing process. *Additive Manufacturing*, *25*, 151–165. https://doi.org/10.1016/j.addma.2018.11.010

Seifi, M., Salem, A., Beuth, J., Harrysson, O., & Lewandowski, J. J. (2016). Overview of Materials Qualification Needs for Metal Additive Manufacturing. *JOM*, *68*(3), 747–764. https://doi.org/10.1007/s11837-015-1810-0

Shen, H., Li, S., Gu, D., & Chang, H. (2012). Bearing defect inspection based on machine vision. *Measurement*, *45*, 719–733. https://doi.org/10.1016/j.measurement.2011.12.018

Sitthi-Amorn, P., Ramos, J. E., Wang, Y., Kwan, J., Lan, J., Wang, W., & Matusik, W. (2015).

MultiFab: A machine vision assisted platform for multi-material 3D printing. *ACM Transactions on Graphics*, *34*(4). https://doi.org/10.1145/2766962

Sood, A. K., Ohdar, R. K., & Mahapatra, S. S. (2009). Improving dimensional accuracy of Fused Deposition Modelling processed part using grey Taguchi method. *Materials and Design*, *30*(10), 4243–4252. https://doi.org/10.1016/j.matdes.2009.04.030

Suykens, J. A. K., & Vandewalle, J. (1999). Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, *9*(3), 293–300. https://doi.org/10.1023/A:1018628609742

Tang, P., Akinci, B., & Huber, D. (2008). Characterization of three algorithms for detecting surface flatness defects from dense point clouds. In J. A. Beraldin, G. S. Cheok, M. McCarthy, & U. Neuschaefer-Rube (Eds.), *Three-Dimensional Imaging Metrology* (Vol. 7239, p. 72390N). International Society for Optics and Photonics. https://doi.org/10.1117/12.805727

Tharwat, A. (2018). Classification assessment methods. *Applied Computing and Informatics*. https://doi.org/10.1016/J.ACI.2018.08.003

Watkins, C., & Holloway, R. (1992). *Technical Note : Q-Learning* (Vol. 8).

Westerweel, B., Basten, R., den Boer, J., & van Houtum, G. (2020). *Printing Spare Parts at Remote Locations: Fulfilling the Promise of Additive Manufacturing*. *Production and Operations Management*. https://doi.org/10.1111/poms.13298

Wuest, T., Weimer, D., Irgens, C., & Thoben, K. D. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. *Production and Manufacturing Research*, *4*(1), 23–45. https://doi.org/10.1080/21693277.2016.1192517

Yadroitsev, I., Krakhmalev, P., & Yadroitsava, I. (2014). Selective laser melting of Ti6Al4V alloy for biomedical applications: Temperature monitoring and microstructural evolution. *Journal*

of Alloys and Compounds, *583*, 404–409. https://doi.org/10.1016/j.jallcom.2013.08.183

Yao, B., Imani, F., & Yang, H. (2018). Markov decision process for image-guided additive manufacturing. *IEEE Robotics and Automation Letters*, *3*(4), 2792–2798. https://doi.org/10.1109/LRA.2018.2839973

Yao, B., & Yang, H. (2018). Constrained markov decision process modeling for sequential optimization of additive manufacturing build quality. *IEEE Access*, *6*, 54783–54794. https://doi.org/10.1109/ACCESS.2018.2872391

Yong, Y. (2012). The Research of Imbalanced Data Set of Sample Sampling Method Based on K-Means Cluster and Genetic Algorithm. *Energy Procedia*, *17*, 164–170. https://doi.org/10.1016/J.EGYPRO.2012.02.078

Zeng, K., Pal, D., & Stucker, B. (2012). A review of thermal analysis methods in laser sintering and selective laser melting. *23rd Annual International Solid Freeform Fabrication Symposium - An Additive Manufacturing Conference, SFF 2012*, 796–814.

Zhang, N. L., & Zhang, W. (2001). Speeding up the convergence of value iteration in partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, *14*, 29–51. https://doi.org/10.1613/jair.761

Zhihua, Z. (2012). Book Review: Ensemble Methods: Foundations and Algorithms. *IEEE Intelligent Informatics Bulletin*, *13*.

Zhou, Q.-Y., Park, J., & Koltun, V. (2018, January 29). Open3D: A Modern Library for 3D Data Processing [Computer Software]. Retrieved from http://arxiv.org/abs/1801.09847

Vita

Born in a small city in China, Rui Li grew up with a happy family. She obtained her bachelor's degree majoring in Materials Science and Technology from University of Science and Technology Beijing. Then she decided to further her education in RWTH-Aachen University in Aachen, Germany. She achieved her master's degree there with concentration of Metallurgical Engineering. Later she came to the University of Tennessee, Knoxville to accomplish her PhD study with Industrial and Systems Engineering as her major. She is grateful for all the supports from her family, friends and advisor.