The University of Warwick

School of Engineering

LAYER-WISE IMAGE EXTRACTION AND CLASSIFICATION FOR METALLIC POWDER BED FUSION ADDITIVE MANUFACTURING USING A COMPUTER VISION MACHINE LEARNING ALGORITHM

A Thesis in

Mechanical Engineering

by

Román Vázquez Lorenzo

Supervised by:

Dr Dmitry Isakov

Submitted in partial fulfilment of the requirements

for the degree of

Bachelor Engineering

March 2021

Acknowledgments

I would like to thank my supervisor, Dr Dmitry Isakov, for his support and honesty throughout the whole project, and for introducing me to the field of Additive Manufacturing. He motivated me to go further and always strive for better results.

I would also like to thank my family and friends, who cheered me up and urged me to overcome challenging and stressful times throughout the global pandemic. Thank you.

Abstract

Methodology and analysis of the development of a software and a machine learning (ML) algorithm for image extraction and classification of a Selective Laser Melting (SLM) additive manufacturing (AM) process is presented. The programmes were created by means of the MATLAB R2020b software package and the video of the surface of the powder bed of the SLM printer was captured using the Xiris™ XVC-1100 smart camera. For all frames of the video captured, mean b^* colour values were obtained and compared, identifying the precise frames required for analysis. An accuracy of 99% was achieved, proving the aptness of the software for Laser-Powder Bed Fusion layer-wise quality monitoring. Thereafter, a Bag of Visual Words (BoVW) ML algorithm was trained using the collected frames, which was subsequently tested, achieving a 92.1% accuracy. Several validation techniques were utilised, confirming software integrity. Finally, the impact of the developed programmes was outlined, focusing on their significance for research and evaluation of the effect of AM process parameters in print quality, and the opportunities for implementation in real-time AM quality monitoring systems.

Table of Contents

Table of Figures

Table of Tables

Nomenclature

AM	Additive Manufacturing	SVM	Support Vector Machines
PBF	Powder Bed Fusion	KNN	K-Nearest Neighbours
EBM	Electron Beam Melting	NN	Neural Network
L-PBF	Laser-Powder Bed Fusion	CNN	Convolutional Neural Network
SLM	Selective Laser Melting	SURF	Speeded-Up Robust Features
LENS	Laser Engineering Net Shape	SIFT	Scale Invariant Feature Transform
ML	Machine Learning	DSLR	Digital Single Lens Reflex
CAD	Computer-Aided Design	DBN	Deep Belief Networks
BoVW	Bag of Visual Words	FNR	False Negative Rates
CV	Computer Vision	RGB	Red, Green, Blue
HSV	Hue, Saturation, Value	BRIEF	Binary Robust Independent Elementary Features

1. Introduction

1.1. Background Information

1.1.1. Additive Manufacturing: Laser-Powder Bed Fusion.

AM is the term that refers to a group of technologies that enable physical parts to be created from virtual 3D models, typically formed layer by layer until the part is finished [1]. AM processes are one of three major types of manufacturing technologies, along with subtractive processes, that minimize the size of the workpiece, and formative processes, that maintain the mass of the workpiece [2]. Having been under development since 1982, over the past three decades the technology has evolved from a tool that is utilised for prototype creation to a viable and robust manufacturing method [3]. This is in part due to a set of unique advantages that it offers over other traditional processes, especially for the automotive, biomedical, energy, and aerospace sector [4]: Complex geometries created through a digital model can be manufactured, material characteristics can be altered for material academics to study, material waste can be highly reduced [5] and the environmental footprint that it creates is much smaller than other industry standard technologies [6].

Dependent on the type of layer-by-layer manufacturing characteristics, AM can be grouped in seven categories: VAT photopolymerization, material extrusion, material jetting, binder jetting, powder bed fusion, direct energy deposition and sheet lamination. As this study will be focusing on metal AM, Table 1.1 shows the three technologies that focus on metal manufacturing, showing the part manufacturing method and the main manufacturers of the metal AM machines. As with most AM technologies, metal AM is being included in critical applications such as implants and aerospace, mainly thanks to its ability to form complex structural shapes [7]. Nonetheless, there are several metallurgical differences between traditional manufacturing methods and AM that must be tackled and improved for final inclusion in critical applications, such as mechanical anisotropy, residual stress, and other common defects [4]. Metallic powder bed fusion (PBF) uses either an electron beam (EBM) or a laser beam (L-PBF) to fuse powder particles on a layer-wise basis [8]. This report focuses on the latter of the applications, particularly a Selective Laser Melting process or SLM, one of the most promising metallic AM technologies due to its ability to create components with a fine surface finish, high density and long fatigue life [9].

Table 1.1. A representation of metal AM technologies showing the abbreviation, full name, part fusion method and common manufacturers (information gathered from [10]).

Powder Bed Fusion		Direct Energy Deposition		Binder Jetting
SLM	EBM	LENS	EBAM	BJ
Selective Laser Melting	Electron Beam Melting	Laser Engineering Net Shape	Electron Beam AM	Binder Jetting
Fused with laser	Fused with electron beam	Fused with laser	Fused with electron beam	Joined with bonding agent
EOS, 3D Systems, Sinterit	Arcam	3D Systems. Voxeljet	Optomec	Sciaky Inc

Fig. 1.1 illustrates the process of a typical SLM printer. The build chamber is an enclosed space usually filled with an inert gas such as argon to produce medical grade products [11]. A three-dimensional Computer-Aided Design (CAD)/STL file is digitally converted into thin geometrical layers in two-dimensions using an AM specialised software. That data is then transferred to the computer controlling the metallic printer, in which scanning parameters and strategies are implemented. Then, a galvanometer mirror-guided laser source (1, 2) scans the powder bed (3), melting selective regions of the surface of the powder bed. Once a layer is finished, the powder bed descends along the Z-axis as much as the predefined layer thickness, as shown by the middle red arrow. After that, the recoater mechanism (4) which may consist of a hard scraper, soft squeegee or roller [12], spreads a new thin layer of powder from the powder feed reservoir (5) over the powder bed. Following this, the laser melts the powder particles again following the defined areas, and the process is repeated until all layers are manufactured. The melted areas solidify in the chamber, and the parts manufactured are extracted. Additionally, the excess powder (6) pushed by the recoater is collected on a reservoir.

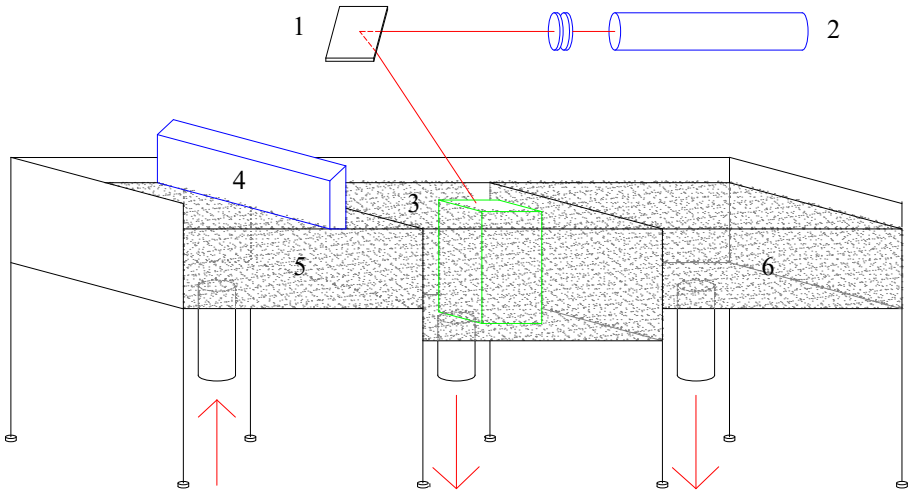

Fig. 1.1. A schematic of a typical SLM machine. Red arrows indicate the direction of the movement of the three powder reservoirs.

1.1.2. Machine Learning

The SLM process operates in a wide range of sizes and time scales. Completion of a single printed part may take over a week and over tens of kilometres of laser pathing [5]. Monitoring of such processes is essential, and as it can be understood, no human can effectively carry out such task, which in addition to taking long hours, it is constantly evolving when new technologies are implemented. Although human operators are obviously unsustainable, ML comes as a great solution for managing these operations.

The study of algorithms and techniques for automation of solutions to complex problems such as this one is defined as ML [13]. These algorithms are able to identify non-linear relationships in large amounts of data [14], and the structure and architecture of these models has been greatly refined over the last decades. Most ML problems can be classified into one of these groups: classification, grouping sets of data into categories; clustering, dividing datapoints into groups with characteristics in common; and prediction, forecasting future values based on historical data [15]. To solve any given problem using an ML algorithm, seven steps must be followed: Data collection, data preparation, model selection, model training, model evaluation, parameter tuning and making a prediction. This report encompasses the creation of a ML model for classification, following all steps for algorithm creation. The problem to solve arises from the need for real time monitoring of SLM processes, and the necessity to study how SLM parameters affect the quality of metal parts printed.

1.2. Project Aims and Objectives

The aim of this study is to develop a ML algorithm for the classification of layer-wise imagery of the AM of metallic pieces in a PBF machine. Real time monitoring of the AM process is crucial for the further inclusion of the technology in the major manufacturing industries, and so is the analysis of the effect of process parameters in part quality. To accomplish the former, a fast and accurate extraction of the desired frames from the Computer Vision (CV) sensors implemented in AM machines must be performed. To accomplish the latter, an accurate classification of the layer-wise images extracted must be implemented. The study hereof presents and evaluates a solution for each of these two undertakings, which were created following the objectives presented below:

1. To automate the extraction of desired frames of a video of the AM process.
    1.1. Create software for the identification of each layer in the AM printing process.
    1.2. Produce a database of training and test data for creation of classification algorithm.
    1.3. Discuss the advantages of the automatic extraction of desired frames.
2. To generate a supervised algorithm for the classification of the images in the database.
    2.1. Select and train machine learning model.
    2.2. Evaluate the trained model using the testing dataset.
    2.3. Analyse the accuracy of the algorithm and discuss further improvements.

1.3. Duration of the Project

The project started in October 5th 2020, and finished March 18th 2021. Prior to the start of the project, the author completed an 11-week long Machine Learning course delivered by the University of Stanford on the online course platform Coursera[16]. This, in addition to past and current Mechanical Engineering education imparted by the School of Engineering at the University of Warwick, and guidance from the project supervisor, enabled the author to complete the project following a structured plan.

1.4. Report Structure

This technical report is organised around the two objectives established above, which were given the shorter denomination of: 1) Image database collection 2) Sample classification algorithm. Most chapters make a distinction and analyse both sections. A brief overview of each chapter is provided below:

1. Section 1 introduces the reader to AM, SLM and ML. This is essential for understanding the motivation of the study and the concepts here presented. The main aim of the project and the objectives established are also stated.

2. Section 2 describes the theory behind ML algorithm development, presenting the algorithm chosen for the development of the study, a (BoVW).
3. Section 3 reviews the publications in relation to the present work up to the date of writing, highlighting the necessity of the development of a fast and accurate algorithm for image classification to utilise in AM monitoring studies.
4. Section 4 describes the experimental setup utilised for video acquisition and the methodology followed for achieving the objectives aforementioned.
5. Section 5 presents the results obtained. The scatter plot, confusion matrix and training and testing accuracies are presented, establishing the robustness and validity of results obtained.
6. Section 6 discusses the results presented. Relevance for future studies, an evaluation of the robustness and validity and an examination of future improvements is done for both objectives.
7. Section 7 summarises the concepts discussed, and a statement of implications derived from the study and recommendations for future work is done.
8. In addition to the chapters, additional relevant information is provided in the Appendix.

2. Theory

2.1. Machine Learning for Computer Vision

CV seeks to understand and analyse digital images and videos in the same way that human vision does [17]. The synergy between ML and CV aims to perform automatic analysis of images and videos to get closer to a human-like vision. The main tasks performed by the combination of both technologies can be divided into three: object classification, object detection and object instance segmentation. This report focuses on the first of the tasks, examining an image to define and categorise the information present in it.

The ML approaches to realising object classification can be broadly separated into two: supervised and unsupervised learning [18]. The differences between the two lies with the labelling of the dataset utilised. Supervised ML algorithms use a labelled dataset that provides meaning to the data and allow labelling of new unlabelled data. On the contrary, the dataset utilised for unsupervised learning is not labelled, and the objective for this ML method is to predict unknown patters in datasets. A more in-depth explanation of common supervised and unsupervised algorithms for object classification is provided hereafter.

2.1.1. Supervised Algorithms for Classification

Rebala *et al* define classification as *"identifying the category to which an input belongs to among a possible set of categories"* [13]. A supervised ML algorithm employs a correctly labelled dataset to identify the group of the input data, and the model that achieves this is defined as the classifier. A classifier can be binary, meaning that the input data can be categorised to two output categories, or it can be multi-class, in which multiple categories can be the outcome of the classified input data. There are multiple types of ML algorithms that are utilised to solve a classification problem, and they are chosen depending on the characteristics and the relationship between the input data of a database. Two which will be referenced later on the report are described hereafter.

Support Vector Machines (SVMs) are popular amongst researchers to solve classification problems [18]. These classifiers create a hyperplane boundary of N - 1 dimensions for n-dimensional feature vectors, which separates data points into classes. This line created is used to classify future datasets inputted in the model. Fig. 2.1 shows how the SVM creates the hyperplane boundary to divide the data into two classes, represented by the red line. The advantages of these types of models are that they can operate with a relatively small database and with sparse data [13]. However, for complex nonlinear boundaries, SVMs require data points (feature vectors) to be converted to higher dimensional space, which can be very computationally demanding.

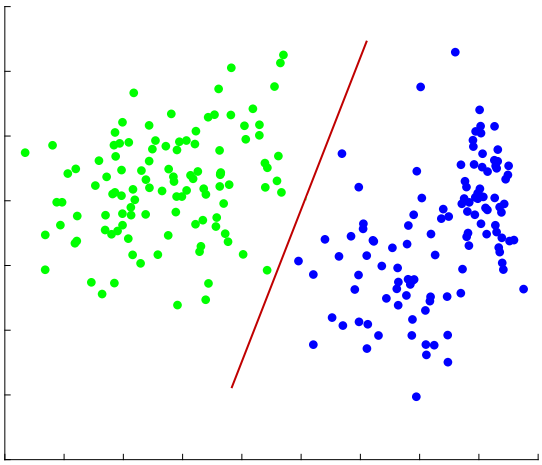

Fig. 2.1. A representation of the boundary created by the SVM algorithm, shown by the red line.

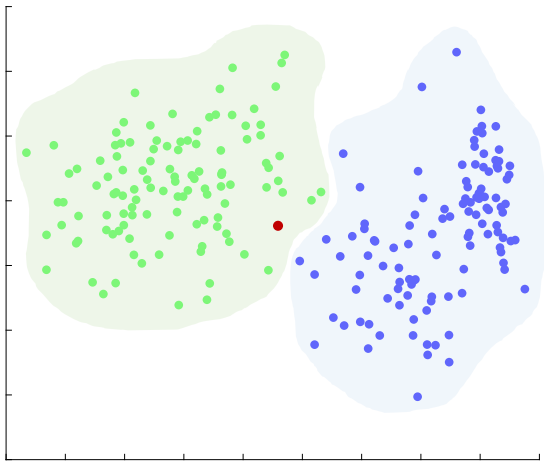

Fig. 2.2. A representation of the KNN algorithm and the classification of a new data point shown in red.

Another popular algorithm for classification is K-Nearest Neighbours (KNN). This classifier is based on the assumptions that similar objects/input data will be close together when plotted on a graph [19]. Hence, the distance between points is measured and data is categorised based on the closeness to other data points. When new data is added, it will be then classified to the nearest class based on its distance to other points. Using the same graph as before,

Fig. 2.2 shows the clusters that the labelled data create, and the red dot exemplifies the addition of a new unlabelled data point. It can be observed that using KNN, the point will be classified to the class represented in green. Notably, this type of classification of grouping elements close to each other is denoted as clustering. However, this type of algorithm is not useful when dealing with skewed datasets, as one of the clusters would always be larger and most new data would be allocated to it.

2.1.2. Unsupervised Algorithms for Classification

Unsupervised learning algorithms are usually employed to identify trends and hidden patterns on large datasets [15]. Nonetheless, when presented with images that can be grouped by similarity, unsupervised algorithms can identify these similarities and automatically divide the dataset into different clusters. This can be thought as a classification method and is usually applied when groups of a large dataset of unlabelled feature vectors want to be identified. In this report, an unsupervised learning method will be utilised, denominated as k-means clustering, hence it is important to describe it for future reference. Similar to the KNN method, the K-means algorithm segregates data into clusters. In this case, unlabelled data is grouped into K clusters, for which the mean value of each is representative of the group and it is identified by being the centre of these clusters [13].

The process to achieve this is described as follows. For a given unlabelled dataset, the aim is to identify the point that represents each K cluster of interest. To do so, the algorithm first randomly selects K points, which will initially represent the clusters. Then, each data point will be assigned to the representative points that is closest to them. From this, the geometrical centre of each cluster is calculated, and it will represent the new cluster centres. This process is repeated over and over again, until the cluster centres do not vary since the geometrical centre of all points in the clusters match the current representative point.

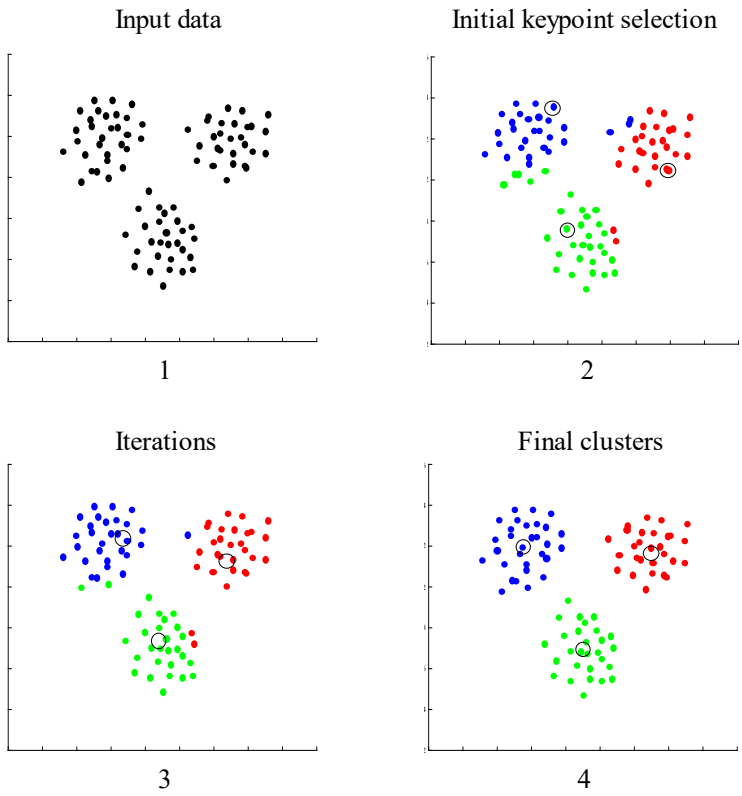

Fig. 2.3. A step-by-step representation of the k-means unsupervised ML algorithm.

2.2. Supervised Classification Models

Having described the supervised and unsupervised classification algorithms that will be referenced further along with the report, this section presents two other classification models that are commonly use in image classification, and which were compared before choosing a model to implement in the study.

2.2.1. Convolutional Neural Networks

Multilayer perceptrons or neural networks (NNs) are supervised ML algorithms inspired by the brain's neuron networks, hence the name. They learn by using interconnected nodes and neurons, breaking down the input in multiple layers for analysis [20]. Convolutional Neural Networks (CNNs) are a special type of NNs which use multiple layers of convolution pooling to identify the degree of overlap between objects of feature vectors [13]. A diagram of the stages of a CNN is presented below, from the convolution and pooling of the images to its implementation to a neural network formed by interconnected nodes.

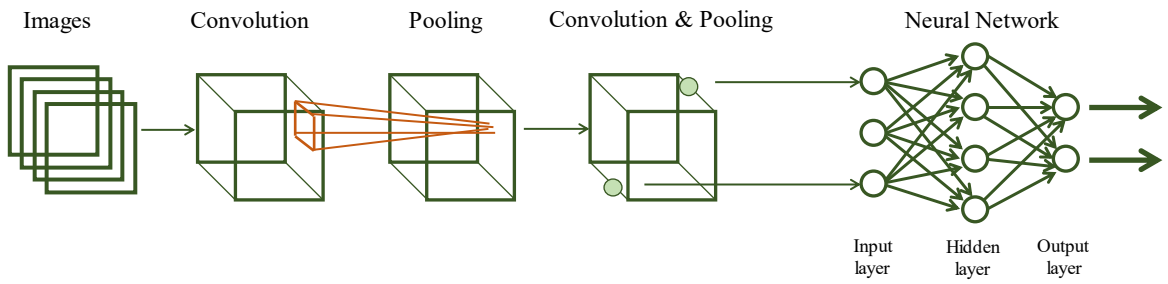

Fig. 2.4. A representation of the structure of a Convolutional Neural Network.

These ML algorithms have gained increasing attention for image classification in the CV community, as they are able to provide exceptionally accuracies while simplifying the algorithm structure and improve training speed. This is described further in the Literature Review section, making a remark on CNNs for metal AM solutions. However, they require large datasets of images for creation, usually in the 10^5 – 10^6 range [21]. That is why for the author's dataset of around 600 images, these models cannot be effectively implemented.

2.2.2. Bag of Visual Words

The use of a BoVW for the description and classification of images has been widely covered in the literature [22, 23], as it is flexible, efficient and thorough. Inspired by the classification of writings based on the frequency of each word, the categorization of images is performed based on the *k* most representative features of an image. The steps to classifying a set of images into several groups are the following:

1. Identifying and describing areas of concern. The Speeded-Up Robust Features (SURF) [24] is one of the most used methods for extraction of local features for its speed, and has become a *de facto* standard [25].
2. Visual words vocabulary generation. The *k* most representative samples obtained in step 1 are selected to create a visual word vocabulary. The most commonly used approach for selecting the most significant samples is through a *k-means clustering* algorithm [22].
3. Creation of histograms for each image according to the frequency of visual words. Based on the similarities in the histograms created, the images are compared and clustered into the main labelled groups.

This method provides the author a flexible approach to image classification. With the objective being a fast and accurate image clustering technique, the BoVW was explored. Additionally, as the training and testing set of images is relatively low in comparison to the benchmark for training of a ML algorithm and they were to be classified in only three different groups, other more complex methods such as a Convolutional Neural Network were deemed inadequate. Fig. 2.5 shows a

flowchart of the BoVW process explained, and a more in-depth explanation of the available methods for feature extraction (step 1 of the BoVW classification process) is done hereafter.

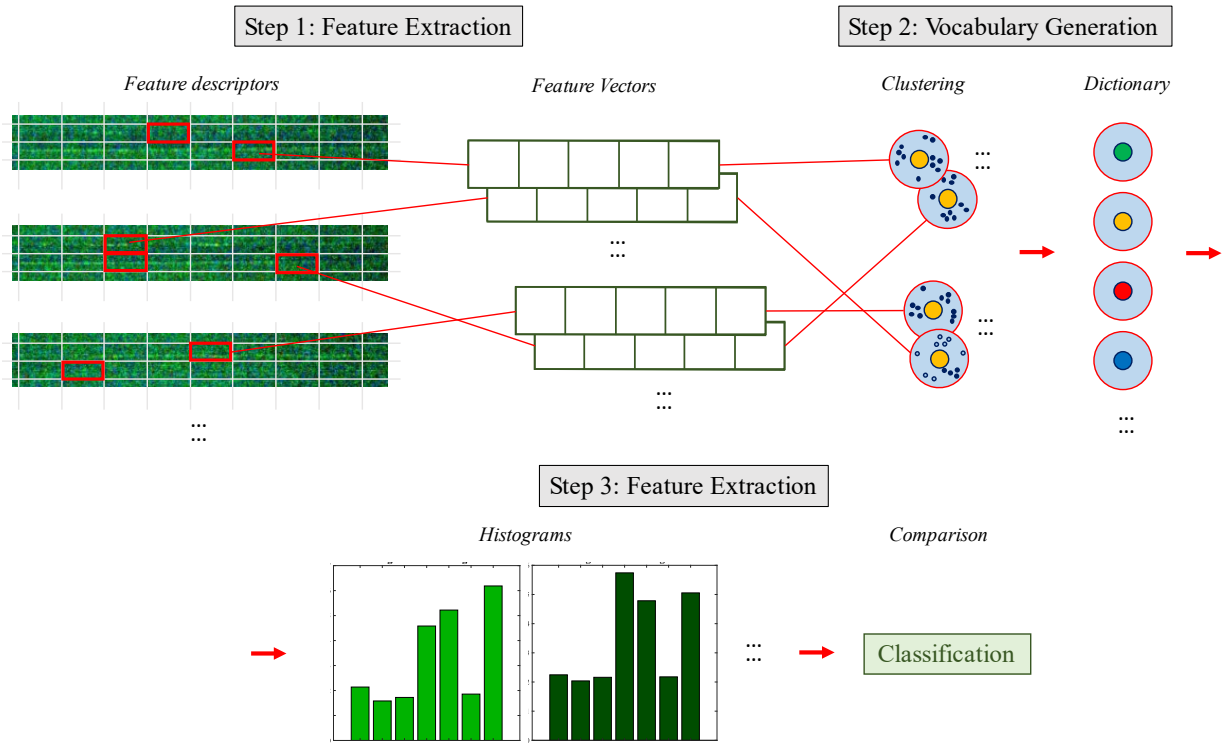

Fig. 2.5. Flowchart of BoVW process, divided into the three main steps for classification.

2.2.3. Feature Extraction

In order to generate a visual vocabulary and histograms for classification of images, the main features of all data samples must be extracted. The process of feature extraction is comprised of two steps: feature detection, in which the images are divided into patches to represent the most informative sections of the images and obtain the appropriate keypoints, and feature description, in which a descriptor is computed for every patch and contains information of every keypoint across all images [26]. Both these processes can be accomplished using what is called a feature extractor, and the most prominent and studied ones are described below.

In 2004, Lowe *et al.* developed the Scale-Invariant Feature Transform (SIFT) descriptor [27]. It derives a feature descriptor from the relative orientation and gradient magnitude of neighbouring pixels. Around each keypoint, four 4x4 subregions of a 16x16 region are created, each forming an 8-dimensional histogram. This means that the descriptor is 4x4x8 = 128 dimensional. While this method is highly discriminant, being a 128-vector renders it slow to compute and match.

Fig. 2.6. A visual representation of SIFT keypoint selection and feature description.

One of the most well-known descriptors is the Speeded Up Robust Features (SURF), which was developed by Herbert *et al.* in 2008 [24]. The objective of the creation of the descriptor was to speed up the feature extraction process from the previously developed SIFT. To do this, it relies on integral images [2], since they allow for efficiently generating the sum of the values of a grid. A blob detector based on the Hessian matrix is used to identify the keypoints, which measures local change around each point and the keypoints are selected where the change is the largest. This can be performed quickly using integral images. Additionally, an upright SURF descriptor was later developed [28] which is non invariant to orientation, hence is even faster for still images or videos which do not require analysis of the orientation. While the descriptor also allows several parameter settings, the 64-vector feature detector has become the standard, resulting in 256 bytes necessary for representation. This method highly increases speed and robustness of the descriptor in comparison to the SIFT descriptor, and since its publication it has been used for many image feature extraction applications [26].

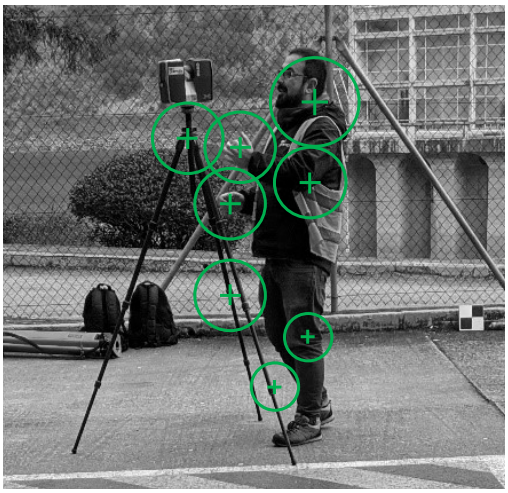

Fig. 2.7. A visual representation of the SURF descriptor. This descriptor first evaluates the circumference surrounding the keypoints to then establish a square integral image around the keypoint. Image adapted from [29].

In 2010, Calonder *et al.* [30] presented the Binary Robust Independent Elementary Features (BRIEF) descriptor, with the aim of producing a faster and less computationally intensive descriptor than both the ones described above. To do so, several pairs (128, 256 or 512) are selected randomly in patches around keypoints. Then, the intensity difference is computed in these pairs and in the case that the intensity of a location is larger than at another, it returns a 1. While this descriptor is faster than the SIFT and SURF described above and is 32-dimensional, hence less computationally intensive, it has been less studied for applications and the implementation in common programming languages such as Python or MATLAB is more complex. Knowing this, the author concluded that the SURF descriptor allowed for the most flexibility and simplicity out of the other analysed, in addition to rendering great speed results.

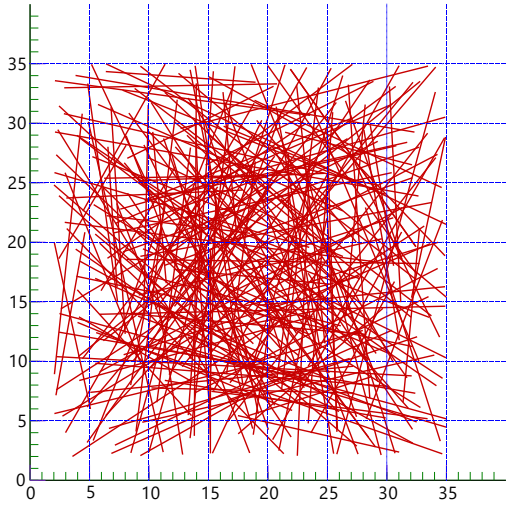

Fig. 2.8. An example of random sampling pattern using the BRIEF descriptor. Image adapted from [30]

To summarise, having understood and analysed the main approaches to CV image classification using ML, the author was able to establish the algorithm and model to develop. Hence, knowing that the dataset to be analysed was relatively small and labelled, and the objective being the creation of a simple yet efficient algorithm, a supervised ML algorithm in the form of a Bag of Visual Words was selected, and training methods using SVMs and KNNs were analysed. This method also requires understanding of K-means clustering, a common unsupervised learning algorithm. Moreover, studies up to date of publication are shown below, which also constituted an important factor in algorithm selection.

3. Literature Review

Heavily regulated industries like the automotive, aerospace, and aircraft sectors require a high degree of component quality, part reliability and reproducibility [12]. SLM is a promising technique for the production of complex metallic products, and extensive research focus is being put on the avoidance of common defects in part production [31]. Several factors have been considered to be the causes of part flaws, such as insufficient supports, non-homogeneous powder deposition, ineffective heat exchange, and inadequate process parameters [32, 33]. The consequences of key process parameters like laser power, melt pool geometry, scanning speed, powder temperature and layer thickness are described in [4]. When it comes to L-PBF, parameters associated with laser behaviour are recognised as the most influential in part quality of the finished product, specifically laser power, scan speed, scan spacing and bed temperature [11, 34, 35].

Adapting laser parameters can help improve the quality of the part and reduce common defects, as the laser influences the behaviour of powdered particles. Debroy et al. [4] groups common anomalies in piece formation into four: loss of alloying elements, porosity and lack of fusion defects, surface roughness, and cracking and delamination. The appearance of external defects translates into a need for post-processing techniques, increasing cost per part manufactured [8]. However, the largest challenge comes from the emergence of internal defects as, in addition to being expensive, it renders post-processing inspection methods unattainable [32]. As such, early prediction and avoidance of defects is necessary to meet industry requirements, hence numerous monitoring techniques have been explored in recent studies. Grasso et al. [36] classifies these process defects and presents several monitoring methods that use pyrometers, regular and thermal cameras and acoustic sensors to evaluate the effect of process parameters in part quality. The systems utilised are divided in co-axial and off-axial, based on whether the sensors are placed on the optical path of the laser or not [37].

Co-axial technologies usually employ a combination of pyrometers and thermal camera to measure and analyse the melt pool size, shape and temperature profile [38-42]. Alternatively, using interferometric imaging, Kanko et al. [43] and Neef et al. [44] were able to analyse scan path geometry and surface patterns, respectively. When it comes to off-axial systems, researchers often use digital cameras to analyse the formation of the manufactured piece layer by layer [45-50]. Using additional illumination, 2D frames of each layer are captured to analyse surface patterns on scanned slices. In addition to that, several researchers include a projector to implement fringe

projection, capturing 3D imagery to analyse topography changes in the build plate [48-50]. The report here presented utilises an off-axial method which will be later described in section 4.1.

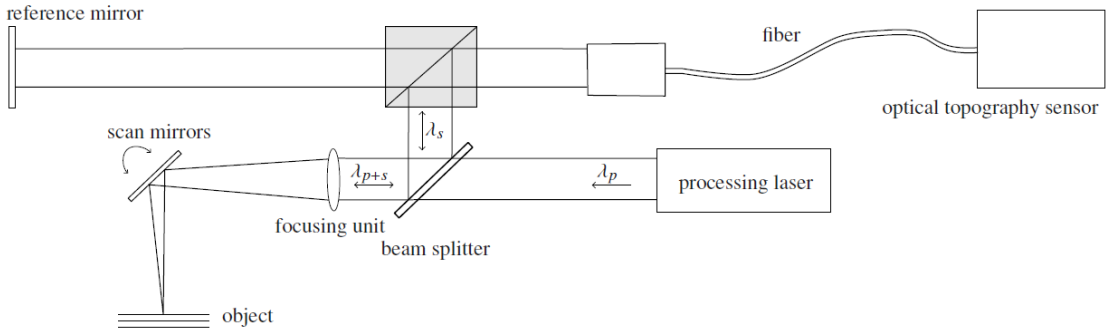

Fig. 3.1. An example of a co-axial technology, showing the integration of an optical topography sensor to the SLM process, developed by Neef *et al.* [44].

3.1. Feedback control for L-PBF manufacturing and shortcomings

Around half of 108 manufacturers who completed a survey for the multinational consulting firm PwC state that the uncertain quality of the final product is the top barrier for AM adoption [51]. While most of the research to overcome this has been focused on identifying the key parameters that affect part quality, few studies have utilised the knowledge acquired to create real-time feedback control methods [3]. Mercelis et al. [52] installed a video camera and a photodiode on an SLM machine to monitor the melt pool area during the printing process, and the data obtained from both sensors served as input for a control unit that maintained melt pool area to desirable values. Similarly, Craeghs et al. [40] determined the relationship between the laser power and a photodiode sensor and created a real-time control loop to adjust processing parameters based on the melt pool area. Additionally, SLM machines manufacturers are attempting to incorporate control technology to their instruments. 3D Systems Corporation, a large American AM machine manufacturer, has equipped each laser of their newest metal printers with an optical train that monitors melt pool geometry in real time [53].

The lack of research towards feedback control for metal AM and L-PBF particularly implies that real-time process monitoring and analysis is still in the early stages, and several shortcomings are rendering the process slow. Yadav et al. compares current monitoring systems available to a “black box” [54], meaning that data acquired from the printing must be post-processed for analysis. This greatly limits the value obtained from installed sensors and takes away from the idea of real time monitoring. Additionally, studies focusing on identifying accurate parameters, are failing to provide a generalised method for L-PBF monitoring. This clearly comes into play when introducing new materials to the printing process, as the results obtained and developed

are not applicable to the new geometric characteristics [34]. On top of that, although melt pool morphologies have risen to be the most studied parameters for defect detection due to the dependency of melt pool dynamics on the quality of the part [38, 40-42, 55, 56], this method is restricted to specific material parameters, hence excessively time-consuming and expensive when introducing new materials.

3.2. Machine Learning for defect detection in metal AM

Through ML algorithms, overcoming challenges associated with understanding part quality in metal AM from process parameters becomes a reality. Data-driven methods evolve and improve as more and newer training data is provided [57], hence limitations related to over-individualisation may be partially solved. Additionally, as ML makes real-time control quicker and more effective, post-process inspection costs can be reduced [58]. For these reasons, in-situ process monitoring technologies are rapidly growing and so are studies on the matter. Data collection, data pre-processing and preparation, model selection, model implementation and model evaluation are the main steps for creating an ML algorithm [59], and so researchers have centred their studies on analysing the different options for each step of the process. The following paragraphs presents several of those studies, paying particular attention to the data pre-processing methods used and the model implemented.

Information gathered from digital images or videos can be analysed through CV algorithms. Many monitoring sensors capture images of the build process layer-by-layer; thus many ML approaches utilise image data for analysis [14, 57, 60-65]. Petrich et al [64] implemented into two supervised ML algorithms, an SVM and a NN, for classification of feature defects. Both image data from a CT scanner and from a DSLR camera taking photographs of each layer, were labelled and processed to match each other, and then both algorithms were evaluated using cross-validation obtaining close to 90% accuracy on the NN for anomaly detection. Building upon this study, Gobert et al. [62] explored the possibility for improvement of the SVM classifier. Through increasing sample size and using an ensemble SVM classifier allowing for more sensor information to be fed into the algorithm, such as input images with different lighting conditions, the accuracy of the model increased from 65% to 85% for classification of anomalous and nominal image frames.

Scime et al. [57] developed an algorithm for anomaly detection using only standard hardware of an EOS M250 L-PBF machine [66] with the intent of reducing complications and costs of the in-situ monitoring process. The model chosen was a supervised ML algorithm known as bag-of-keypoints or words and was implemented using MATLAB. It was used to identify and classify six different anomalies (recoater hopping, recoater streaking, debris, super-elevation, part failure and incomplete spreading). The total accuracy of classification was estimated as 95%, however

the algorithm had difficulty in detecting recoater streaking with a 50.6% accuracy. In addition to traditional image data, several other studies have focused on thermographic imaging. Baumgartl *et al.* [14] put together a small model in terms of computational costs using image data obtained from an off-axis thermographic camera and a CNN. This model does not require extensive data pre-processing hence can be applied to other data formats and achieved a defect detection global accuracy of 96.8%.

Aside from CV algorithms such as the ones described above, alternative data acquisition approaches have recently been explored. Acoustic and photodiode. Shevchik *et al.* [67] recorded acoustic signals using a fiber Bragg grating sensor during a PBF AM process in which intentional manipulation of process parameters was done to form pieces of different porosity, establishing high, medium and low part quality. These time-domain signals were transformed into the frequency domain to train a spectral CNN. Through this method, an accuracy of 83-89% was obtained for part quality differentiation. Similarly, Ye *et al.* [9] captured acoustic signals using a microphone, and trained a deep belief network (DBN) to classify recorded data into four different part anomalies: balling, slight balling, overheating and slight overheating. This study utilised unprocessed signal data while achieving a 72.43% accuracy, proving the possibility of using raw data and minimising cost and time in data pre-processing procedures. Lastly, Okaro *et al.* [34] took another approach to reduce cost and time of part certification and pre-processing. Through the implementation of a semi-supervised gaussian mixture model, the authors were able to make use of the large amounts of unlabelled data that is usually discarded when training supervised algorithm. The data was obtained from a photodiode place in the L-PBF machine, and a 77% classification accuracy was stated from cross-validation and comparison to CT results.

3.3. Image Retrieval and Classification Algorithms

Having reviewed the most recent literature in the advancements of the creation of feedback control systems for AM, the author has observed that researchers have focused on the detection and classification of defects during the AM printing process. However, this is only one of the steps necessary for achieving such monitoring technology. To detect defects on the printing process, a dataset of images for analysis has to be first gathered. Additionally, to establish the relationship between specific process parameters and common defects, the parts printed have to be classified and labelled to the particular input parameters in real time. The author has not found research papers that focus on these two stages of the path to achieving real-time monitoring systems for L-PBF AM systems. Nonetheless, extensive research has been done in image retrieval and classification algorithms in other industries, the technology being in a mature state in some of them [17]

Machine vision for surveying purposes has been widely implemented in many domains [68]. In the agricultural sector, ML algorithms for weed detection has been of paramount importance for reducing yield loses. Recently, Abouzahir *et al.* [69] have developed a classification and detection algorithm by combining a BoVW and an NN which is able to discriminate between weeds and crops in real time at an over 90% accuracy. In the medical industry, Wei *et al*. [70] use a Multi-scale CNN to identify and define lung nodule characteristics, to then classify them into benign or malignant nodules. And in the topic of environmental safety, Pang *et al.* [71] propose a layer-by-layer classification algorithm for risk classification and single-environment risk. This allows to categorise and predict the probability of social risk, as well as judge the level of real risk that a person is subject to. From these studies, it can be noted how valuable image monitoring systems is in all areas of human work and interaction, and their effectivity at solving real problems motivated the author to further advance the implementation of CV and ML systems into metal AM processes.

3.4. Contributions of Present Work

Having understood the need for metal AM feedback control and the value that ML provides to solving CV problems from the review of past research studies, this report focuses on the implementation and development of two pieces of software: one that is able to automatically gather the desired layer-by-layer frames in real time, and another one that labels the layer-by-layer images to recognise which sample associated with a specific set of process parameters said images belong to. Through this study, it is shown that a supervised classification ML algorithm of layer-by-layer images of surface patters in metal AM printed samples can be implemented to accelerate the development of real time feedback control systems for AM machines. The implementation of the software developed in the study has the potential of reducing the costs incurred by post-processing techniques, as real-time quality monitoring and the analysis of the effect of process parameters in part quality can be done. Ultimately, the present work has the objective of increasing the chances of L-PBF manufacturing in entering major manufacturing industries, provided cost-effective piece machining and enabling complex geometries to be developed at a competitive price.

4. Experimental Design and Methods

4.1. Experimental Setup

The experiment and data collection have been conducted as follow. The three samples in form of rectangular prolongated blocks 2x100x8 mm^3 were fabricated in a single operation run using EOS M250 [66] metal SLM printer. Fig. 4.1 shows a mutual arrangement of the samples on the build plate. The samples were fabricated from a commercial Solid Steel SL318 powder feedstock.

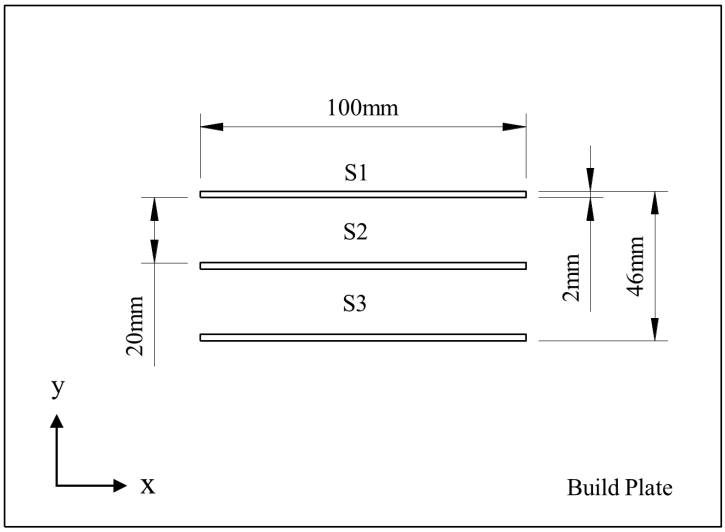

Fig. 4.1. A schematic of the arrangement of samples on build plate.

A simple geometry was chosen establishing an in-control environment to avoid excessive defect formation. This improves clarity of the results obtained and reduces the likelihood of random errors and inaccuracies occurring. Future research will be aimed at extending the study to more geometrically complex parts. To provide a difference in the processing and microstructural properties, each sample was fabricated using individual processing parameters. Table 4.2, lists the machine settings for laser energy and scanning velocity for each sample. This resulted in distinct features for each, permitting the creation of the image-classifier based on part characteristics and the evaluation of the effect that different laser parameters have on part quality.

Table 4.1. Parameters of video acquisition using Xiris™ XVC-1100 camera.

Frame rate (fps/s)	Spatial Resolution (pixels)	Megapixels (MP)	Pixel size (μm2)	Bit Depth (bits)	Colour Filter Array
55.0	1280 (H) x 1024 (W)	10.0	6.80	8.00	Bayer RGB

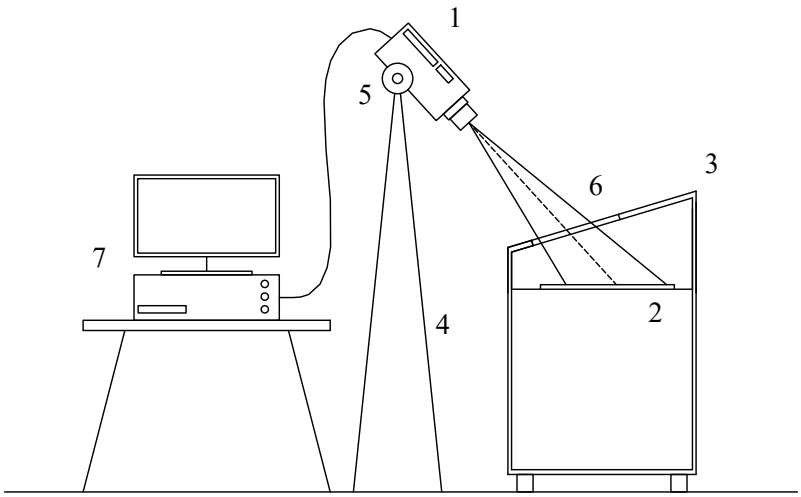

Fig. 4.2. A diagram of the experimental setup diagram. Side view of printer, camera and control system.

Fig. **4.2** shows the experimental setup used for the acquisition of the video for analysis, consisting of a Xiris™ XVC-1100 smart camera (1) which records the SLM build process of the powder (2) happening inside the build chamber (3). The camera is supported by a tripod (4) and set at an angle Φ using the tripod mount (5). It records through a window (6) that encloses the building chamber. This reduces video quality and overall brightness; thus, an additional light was added. The Xiris™ XVC-1100 camera [72] was originally designed for recording welding and laser processes. It is capable of recording colour video of a High Dynamic Range of 140+dB which allowed the author to obtain unsaturated images of the laser. A 50mm focal length 10MP Kowa™ lens [73] was attached to the camera body and was set to and aperture of 2.8 and focal length of 0.2. A PC and monitor (7) were connected to the camera using an adapter, and the Xiris WeldStudio™ software was used for image display and parameter control. Table 4.1 shows the main parameters of the camera and the video.

Table 4.2. Parameters of EOS M250 printer for sample printing.

Powder	Laser Energy (J)			Laser Speed (mm/s)			Layer Thickness (µm)
Stainless Steel 316L	S1	S2	S3	S1	S2	S3	40
	0.80	0.90	0.70	1.50	1.00	0.75	

4.2. Image Database Collection

The image acquisition setup captured the full SLM process, resulting in 138.8GB of data in the form of a 90 minutes '.avi' file. From the video, the database of frames for analysis was obtained as explained in the subsequent sections. The data presented in this study is formed by 600 images of the printed parts. Each sample is 8mm in height and the layer thickness was set to 40µm hence, 200 images of each sample were able to be collected.

Since the video recorded contained the full printing process including recoating of the powder after each layer printed, the desired images had to be edited that the final file contains only useful information of the melting process excluding the recoating phase. The process to do so manually would be extremely time consuming and inaccurate. As it is a repetitive process, the author was able to develop a function using MATLAB that reads the video, selects the desired frames based on the difference of colour between frames and saves them on a local folder. The specific code for the function can be seen on the Appendix A: Code for image database collection, and a diagram describing the process is shown in Fig. 4.3.

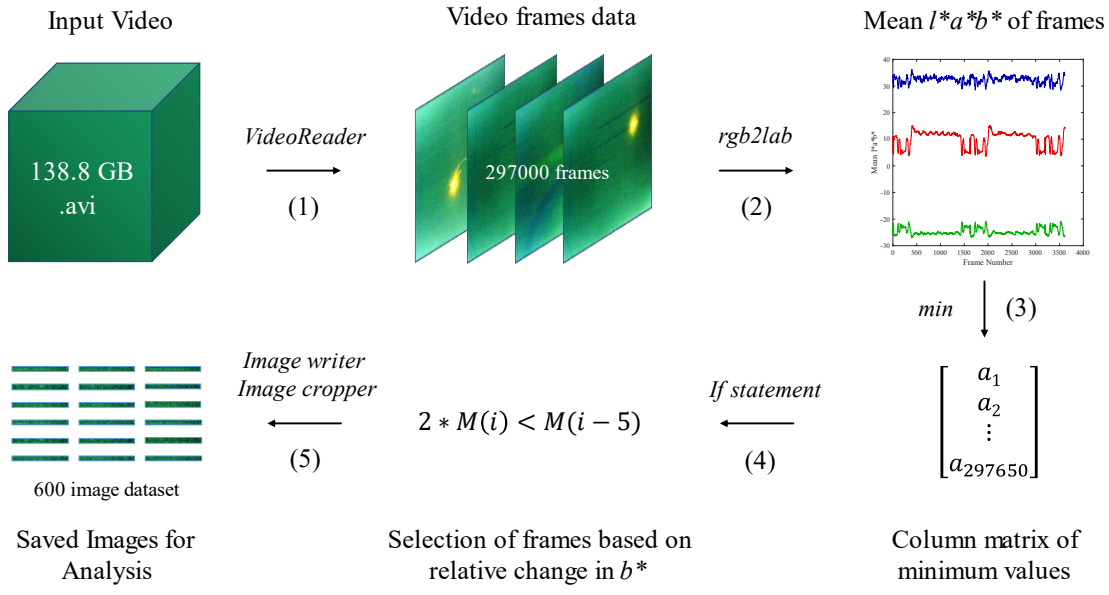

Fig. 4.3. A flowchart of the image database acquisition process.

The programming workflow for image data acquisition and post-processing (editing) is divided to 5 stages. In Stage 1 the video is recorded, for which the object "VideoReader" is used [74]. This object has several properties that contain information of the video file, which can be read and modified. Using one of said properties, a variable for the number of frames is created. Stage 2 converts all RGB frames to the l*a*b* colour space using the object "rgb2lab"[75], a device-independent colour spectrum that covers the entire range of human colour perception [76]. In contrast to the RGB spectrum, the l*a*b* or CIELAB colour space replicates human vision and allows the author to differentiate the frames in which the laser stops running.

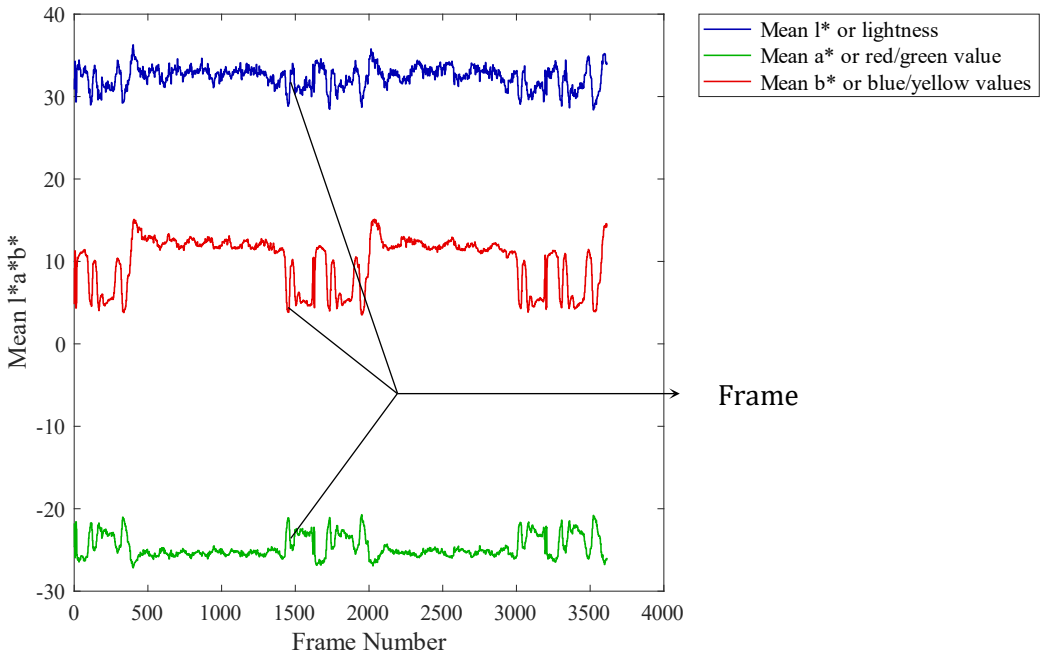

Fig. 4.4. A graph of the mean l*, a* and b* values for the frames in the first minute of the video.

The mean values for *l**, *a**, and *b** are obtained for all frames. Fig. 4.4 shows a plot of said values in the y-axis against the frame number in the x-axis for the first minute of the video. A clear trend can be observed: at the start of the video, values fluctuate, to then stabilise for several frames and go back to rising up and down. This trend corresponds with the operation of the laser, the more stable values associated with the laser being in operation, and repeats 200 times throughout the full video. Knowing this, the instant at which the laser stops operating can be specified, which will correspond with the frame desired for analysis. The author chose the *b** values to do so, since they produce the plot that distinguishes best the frames in which the laser is in operation to the ones that it is not. This can be attributed to the laser increasing the proportion of yellow in the frame and *b** analysing the proportion of blue to yellow in each image.

In Stage 3, the array of b^* values obtained, a loop and the "min" MATLAB operator which returns the minimum elements of an array [77] are used to create a matrix containing the minimum value of b^* between the frame being analysed and the subsequent fifty frames. This returns a 297650-column matrix M which is used in stage 4 of the image acquisition procedure. In Stage 4 the values of the M matrix are compared, and the frames are sorted correspondingly. The frames for which the change in minimum b^* between the frame being read and the five frames prior to it is larger than twice the b^* value of the current frame $M(i)$ are associated to the finish of the laser processing phase and it is consecutively sorted and saved for further analysis:

$$S = M(i) \text{ if } 2 * M(i) < M(i-5) \qquad \textit{Eq. 1}$$

where S is saved frame. Correspondingly, the part of the video that are not associated with laser processing (i.e., recoating phase) is rejected and not further analysed.

Through this method, the sudden difference in average yellow colour can be identified and the desired frames can be singled out from the rest. An example of a frame in which the laser is in operation and another of a saved frame for analysis can be seen on Fig. 4.5. Stage 5 of the feature extraction process involves the deletion of redundant frames from the same layer that are saved in the folder since they also comply with the condition set out above, as well as the cropping of the frames to the desired sample sizes. As only one of the frames of each layer is needed, the author coded a programme that automated the deletion of said repeated frames based on the frame number. In the case that the tens digit of two or more frames coincides, only the first frame would be saved, and the rest would be deleted.

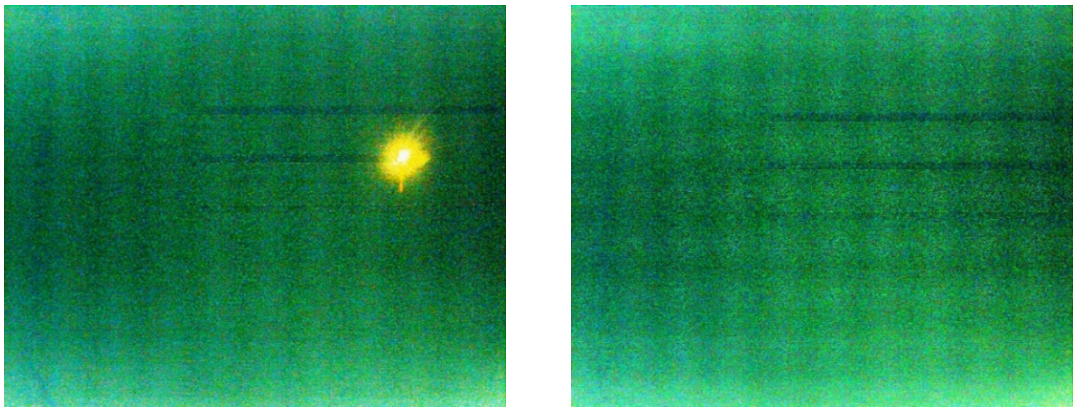

Fig. 4.5. Video frame nº 410 with L-PBF laser in operation (left) and video frame nº 908 saved for analysis (right).

A total of 189 images were obtained from the video with their corresponding frame number, allowing the author to carry out layerwise analysis of the three different samples. Nonetheless, for the creation of a functional classification algorithm, the training and testing database of the different samples was still to be created. To do so, the images were cropped and saved using the

image cropper in MATLAB. Since the placement of the samples and the camera were fixed throughout the video, the x and y coordinates from which the cropping had to be done were obtained for the first image, and all frames were cropped equally, obtaining consistent image sizes. Additionally, the images were labelled by their correspondence to Sample 1, Sample 2 or Sample 3 as shown on Fig. 4.1, completing the pre-processing of images to obtain the training and testing database for further analysis. Fig. 4.6 shows the collected database of images with the three datasets of labelled images according to the sample they belong to.

Fig. 4.6. A schematic of the image database used for the creation of a classification and defect detection algorithm showing the three different datasets of layerwise sample images.

4.3. Sample Classification Algorithm

By analysing the sets of labelled training data obtained as described in section 4.2, a classification ML algorithm was created. Through visual pattern recognition, images from the build process in AM can be assigned and grouped to their corresponding sample. By implementing a fast and low in computational cost classification ML algorithm, the analysis of the effect that process parameters have on part quality can be performed more efficiently. The methodology for building such classification algorithm is described in the following subsections.

4.3.1. Training of Samples

Using the "imageDatastore" object [78] the acquired database of images was stored, gathering all images and subfolders labels. From this, the author allocated 40% of the images to create the training set and the rest allocated to testing. The image partitioner used was set to randomise the selection of images for the testing and training set every time the code was run to avoid systematic errors.

To define the vocabulary for the BoVW, the "bagOfFeatures" object was utilised [29]. It is a flexible object from the MATLAB environment that allows adjustment of feature detector settings, vocabulary size, strongest feature selection and other relevant configurations. A combination of theory (summarised in section 2) and experimental iteration improvement was utilised to select the suitable bag of features configuration, the latter shown in Table 5.1 and the final configuration bolded.

The final bag of features created first selects the feature point locations using the Grid method, from which around 310000 features of the training image dataset of each sample are extracted using the SURF method explained in the Theory section. Then, 80% of the strongest of those features are saved and balanced to reduce the importance of the most prevalent in order to improve clustering. After that, the sample image with the least number of strong features establishes the threshold for feature analysis, in this case 249421 features. Using *k-means* clustering as described in section 2.1.2, the features are grouped in 7, establishing the 7 most relevant visual words, also known as the dictionary, which will be used to describe and compare the categories (S1, S2 and S3).

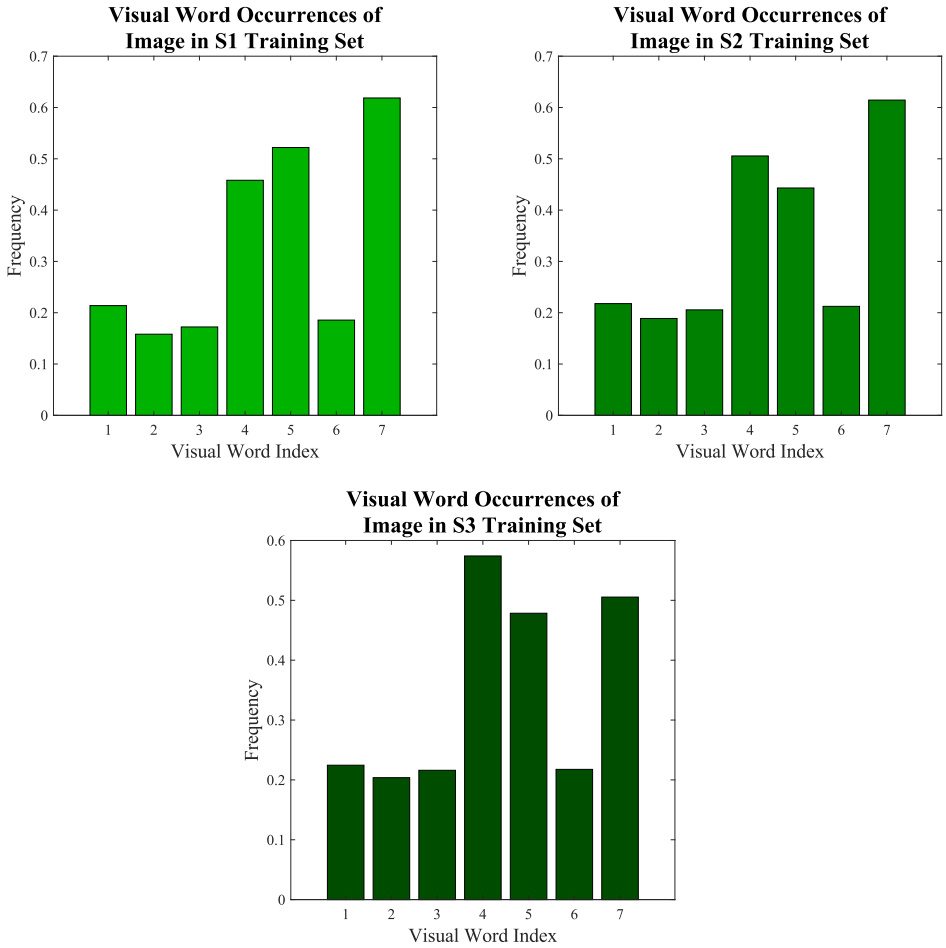

Fig. 4.7. Histograms of Visual Word Occurrence for three different images of the three different sample categories.

This process is passed over every training image, creating histograms of the frequency of the most relevant visual words, shown on Fig. 4.7, which become feature vectors for each image. The y-axis represents the percentage of pixels matched to each visual word, and the visual word index is represented on the x-axis. From the histograms, the differences in frequency of the 7 visual words between the three different samples can be observed: visual word 2 is the most prominent in sample 1, feature 7 in sample 2 and feature 5 in sample 5. These feature vectors are usually referred to as fingerprints, as they identify each image uniquely [57]. A table of 339x8 dimension with all the fingerprints and their associated sample class is created and used to train a classification algorithm using the MATLAB Classification Learner [79]. No subsampling of the training data is performed; hence all the response vectors are utilised for the training. Within the Classification Learner, multiple classifiers were trained, from which the one that provided the highest training accuracy was exported. Ultimately, the comparison of fingerprints of each image allows for the grouping of similar feature data into the established labels (S1, S2, S3). Consequently, using the exported model, future datasets will be grouped into the three different sample categories based on the quantity of visual words observed on their fingerprints.

4.3.2. Validation Techniques

Through the process described above the supervised classification model is created. However, the last step to every ML algorithm is to evaluate its performance and test the validity of the results. K-fold cross-validation, confusion matrices, scatter plots, validation accuracy and testing accuracy are five ways that allowed the author to ensure integrity of the algorithm created. As stated in the previous section, 60% of the images of each dataset were set aside for training of the algorithm, and 40% for testing.

Cross-validation randomly divides the datasets into a training set and a testing set in order to maximise algorithm learning and validity as well as to avoid overfitting and underfitting. The k-fold cross-validation mechanism splits the dataset into k partitions, in which data is distributed at random and the random sets are known as folds [80]. Each partition comprises approximately n/k number of data components, n being the total number of elements and k the number of folds. The author chose 5-fold cross validation, creating 5 randomly generated datasets that improve results validity, and hence having 20% of 60% of the images of the dataset allocated to validation of the algorithm. A diagram of the 5-fold cross validation method is shown on Fig. 4.8.

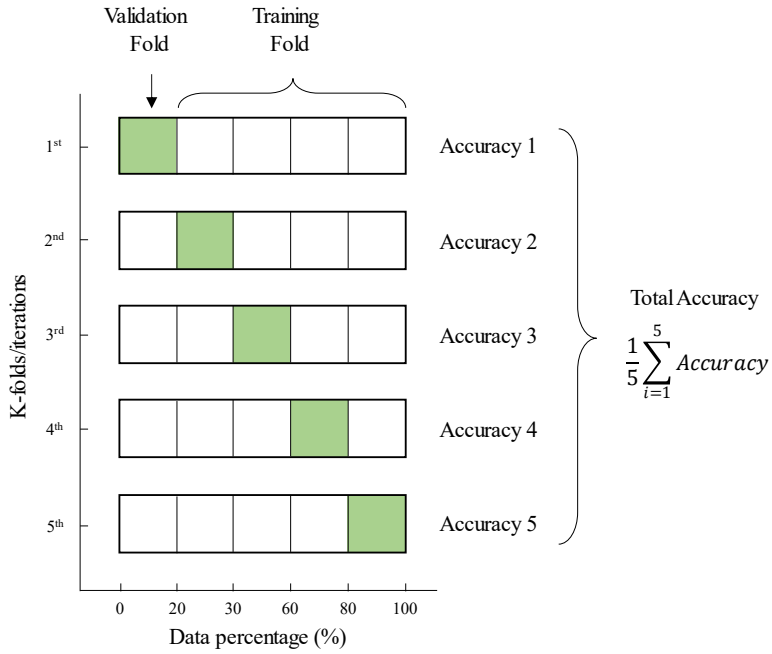

Fig. 4.8. A diagram of the 5-fold cross validation method used to improve validity of the results.

A confusion matrix compares the ground truth labels made by the author to the classification obtained from the ML algorithm. The data used for the comparison must be different than the one used for the training in all executions of a confusion matrix. In the square matrix *A*, the left-to-right diagonal represents the percentage of images that the algorithm has classified in the correct category in comparison to the previously established labels. If the algorithm classifies 100 images of each category correctly, the result would be a diagonal matrix with all diagonal values being 100. Hence, the confusion matrix enables quick visualisation of algorithm performance and it allows the author to identify which categories are mistaken for each other the most, so adjustment in the bag of features and algorithm training can be done in relation to the values shown by the confusion matrix.

Similar to confusion matrices, scatter plots allow the author to visualise the relationships between the data obtained from the BoVW. They are two-dimensional charts that indicate the x and y values of every individual data point, usually in dots [18]. When creating a classification algorithm, scatter plots often illustrate the clusters created from the k-means clustering. Additionally, an x instead of a dot signifies the classified values that were not grouped to the same ground-truth label set by the human, hence show where most of the errors are being made by the ML algorithm. The author also utilised scatter plots to assess whether the model was overfitted or underfitted.

Finally, the validation accuracy and testing accuracy can be obtained from the results of the ML classifier and the matrices and plots outlined above. The former corresponds to the percentage of

images from the validation folds of the k-fold cross-validation that were correctly classified. This value provides an initial estimate of the accuracy of the trained algorithm. Similarly, the testing accuracy corresponds to the percentage of images from the 40% set aside for testing that were classed to the correct label, and provides a real estimate of how accurate the classifier is using new data. The comparison between the two accuracies helps the author determine whether overfitting or underfitting has occurred in the training process, since a large divergence between both values indicates that the classifier created is not valid.

5. Results

5.1. Results of Image Database Collection

To collect the images and create a sample database, the procedure described in section 4.2 is followed. From the analysis of mean b^* values, the author was able to identify the desired frames in the video and save the cropped images of the three different samples. A link to the public folder containing the images utilised is provided on the Appendix. A total of 189 layerwise frames were obtained, meaning that the data base is composed of 189 frames of each sample or a total of 567 images. However, 200 layers of the printing process were recorded hence only 94.5% of the desired frames were gathered using the programme developed. Refer to section 0 for an analysis and explanation of the imprecision.

5.2. Results of Sample Classification Algorithm

As explained on section 4.3.1, to design the most appropriate BoVW classification algorithm and obtain the best results from the algorithm training and testing, several parameters must be specified, which were chosen on a theoretical and empirical basis. From a theoretical basis, the Grid visual feature detector method was deemed most appropriate (refer to Theory), with an 8x8 x-y grid step size in pixels and a block width or patch size to extract upright SURF descriptors of 32, 64, 96 and 128 pixels. Four different square sizes were used in order to extract multiscale features and improve feature selection.

Having done that, the vocabulary size and percent of strongest features to select had to be empirically established. To do this, an iterative approach was performed, in which the bag of features' parameters are tuned according to the values of training and testing accuracy with the objective of obtaining the most accurate and fastest process. In the case that the training accuracy is high, but the test accuracy is much lower, overfitting would be occurring, hence the vocabulary size would be reduced. In the case that the training accuracy is too low, underfitting would be occurring, and the percentage of strongest features selected would be increased as the model is

not capturing the true trend accurately [18]. If neither overfitting nor underfitting were occurring, the parameters would be tuned to obtain the highest training and testing accuracy. Table 5.1 shows the main iterations for parameter tuning.

Table 5.1. A table with the main iterations for classification algorithm parameter tuning.

#	Vocabulary Size	Percentage of Features (%)	Method	Overfitting Underfitting	Trained Accuracy (%)	Test Accuracy (%)
1	20	80	Linear Discriminant	Overfitting	99.4	44.74
2	10	80	Subspace KNN	Overfitting	98.5	34.21
3	5	80	Quadratic SVM	No	91.2	92.11
4	6	80	Fine KNN	No	91.7	92.3
5	6	60	Medium Gaussian SVM	Underfitting	86.4	84.21
6	6	10	Medium Gaussian SVM	Underfitting	81.4	76.32
7	6	40	Bagged Trees	Underfitting	87.0	89.91
8	6	90	Medium KNN	Overfitting	91.2	85.53
Final	**7**	**80**	**Subspace KNN**	**No**	**93.8**	**92.11**

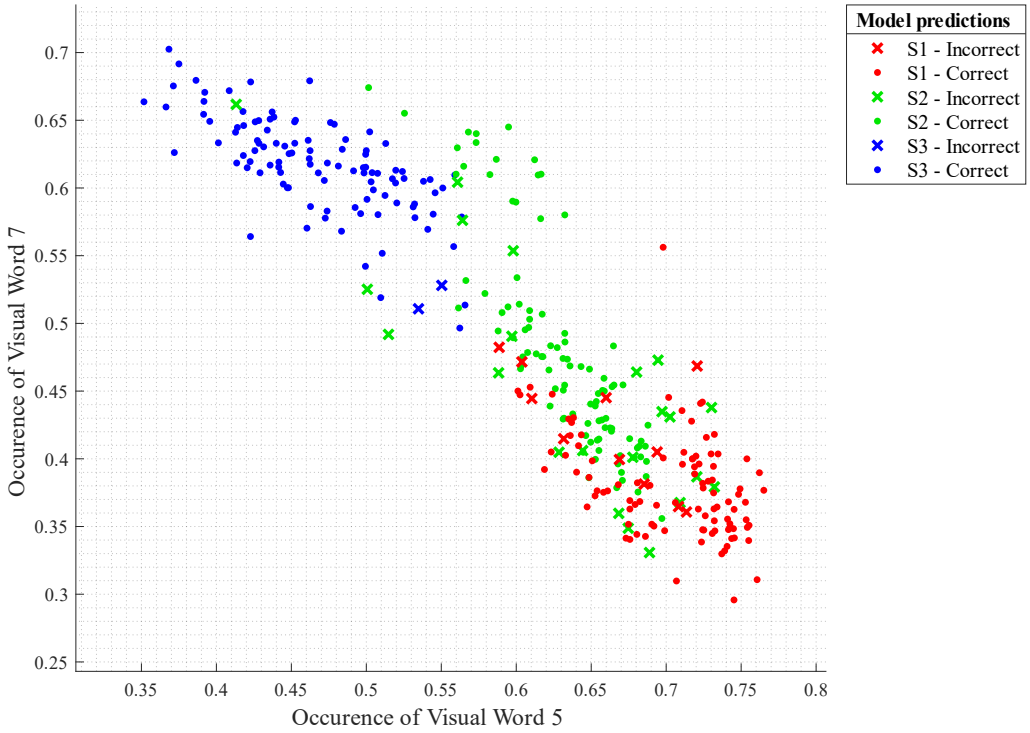

Fig. 5.1. A scatter plot of the relationship between the occurrence of visual word 5 and visual word 7 in all images of the training set, where the misclassified images are displayed with an x marker.

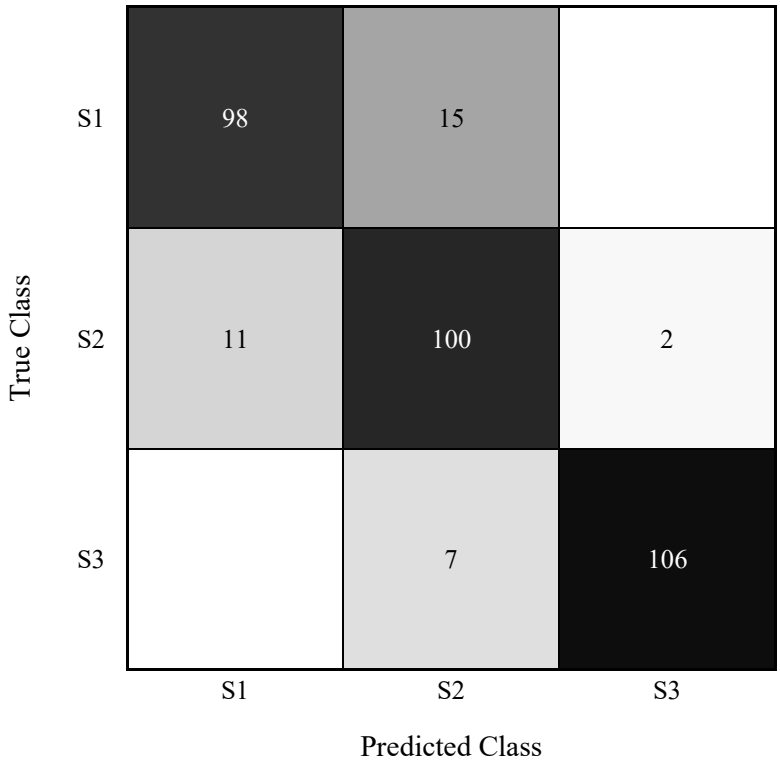

Fig. 5.2. A confusion matrix showcasing the number of correctly and incorrectly classified images. The white-black colour grading indicates the number of images in each box, the darker the colour, the more images in the box.

The last row indicates the parameters that were chosen for the algorithm that was exported. To further validate the results obtained, scatter plots and the confusion matrix of the trained model are shown above. The scatter plot illustrates the relationship between occurrence of visual word 5 and 7, as it provides a clear view of the different clusters. An x marker identifies the images that have been classified incorrectly, allowing visualisation of where the errors are occurring. The confusion matrix also illustrates the classification of the 111 images of each class used for training. The left to right diagonal shows the images that have been predicted accurately, 98 for S1, 100 for S2 and 106 for S3. The other boxes illustrate the images that have not been classified correctly, from which the author can obtain the number of images that were predicted to a class and what class those were falsely been classified to. Additionally, the false negative rates (FNR) for each class can be obtained using the following equation:

$$False\ negative\ rate = \frac{False\ negatives}{False\ negatives + True\ positives} \qquad Eq.\ 2$$

For Sample 1, the FNR is 13.3%, for Sample 2, 11.5% and for Sample 3, 6.20%. This helps the author identify the most probable errors in the classification. Ultimately, the supervised classification ML algorithm achieved a 93.8% validation accuracy and a 92.1% testing accuracy, values which accurately indicate the performance of the model created.

A final important metric for algorithm evaluation is the time spent in testing. More particularly, the elapsed time in creating the BoVW is also an essential metric for parameter tuning and evaluation. With the final parameters established, the former was observed to be 62.99 seconds and the latter to be 10.72 seconds. These and the above-mentioned results will be further analysed and discussed in the following section.

6. Discussion and Analysis

The data and results shown above allow for a comprehensive analysis of both the software developed for image dataset collection and the model created for classification of the images acquired. In this chapter, a discussion of the results, an appraisal of the relevance of the work for future studies, an evaluation of validity and errors and an investigation of further improvements and is done for both sections.

6.1. Analysis of Image Dataset Collection

Chapter 4.2 describes the process for collection of the images required for creation of a classification algorithm. Through the study of the changes in b^* colour values throughout the video

of the printing process, the author was able to create a database of 567 images separated in three datasets containing 189 layerwise images of each of the three samples printed. Interestingly, the author has found little evidence of other software developed for layer-wise feature extraction from a full video of a SLM printing process. This suggests that manual extraction of the desired features from a video is often performed, a time-consuming process and highly cost inefficient. Notably, the acquisition of the desired frames the video did not require large modifications and only a variation of the frame colour space was done, which is not a computationally intensive process. Additionally, the database collected was used for further creation of a ML algorithm, proving the serviceability of the software developed.

It is for these reasons that the author believes that the software developed may be of high value for use in future studies. Other videos recorded from SLM printing processes can be acquired and analysed through the software, obtaining the relevant frames of the samples layer by layer. Most notably, the use of the *l*a*b** colour space for image analysis is highly generic, as it can be applied to multiple types of videos recorded from any camera model. Not only videos recorded in the visible light spectrum, but the software could also be applied to thermal imaging, as the activation and deactivation of the laser in the SLM printing process will always occur and cause a change in *b** values, stabilising these when the laser is in operation.

Although the video utilised in this study is cyclical and all layers have the same characteristics, the software is independent of repetition and can be employed for more complex AM printing. Additionally, the code is highly intuitive and flexible, allowing easy modifications to be made to adjust to changing parameters in future studies. The software created is ready to be employed in ex-situ quality monitoring and post process analysis of the layerwise quality of laser-based AM printing. It constitutes a fast, inexpensive and low computationally intensive method for image extraction. Ultimately, the software has the potential to be utilised for in-situ quality monitoring of SLM printing processes, extracting the desired frames for analysis. However, the accuracy and speed of the image extraction process must be improved for real time in-situ monitoring.

It can be noted that out of the 200 layers of the samples printed, 189 frames, or 94.5% of the images were acquired from the recorded process. Two main factors may be the cause of this inaccuracy. First, a portion of the video recorded was faulty, as the images were black and had no brightness (see Fig. 6.1). This constitutes around 4 minutes of the full 1h and 30min long video, which is the same as 4.4% or around 9 useful frames in total that were not saved. The likely cause of this is an incorrect focus of the build plate. The other factor that may be the cause of not saving the other 2 frames that are missing is attributed to not adhering to the *if statement* conditions (refer to Eq. 1). That is, the minimum *b** value of the frame being analysed and the prior 50 frames,

and the minimum *b** value of the fifth frame prior and the 50 frames before that, is not twice as large. Ultimately, the first factor is a human error that can be easily avoided, while the second factor is an error coming from the software, hence improvement of the software should be done to avoid further issues. Knowing this, the software can be stated as 99% efficient, as 189 out of the 191 frames available to be acquired were in fact saved.

Fig. 6.1. A flawed frame of the recorded video.

Although a highly accurate software was developed, further improvement of the algorithm could increase its robustness and usefulness for other applications. In terms of the utilisation of the *l*a*b** colour space and the analysis of the mean *b** values of each frame, other mean values and graphs could be employed to behave as cross-validation of the frames saved and acquired. That is, implementing additional *if* conditions that apply to the mean *l** and *a** values, hence considering the behaviour of the plotted graphs shown on **Fig. 4.4**. Additionally, other colour spaces were explored, such as the RGB colour model or the HSV (hue, saturation, value); and although the *b** values were ultimately chosen as they provide the clearest graph for analysis, other mean values can be included in the software model through additional *if* conditions highly improving robustness.

In terms of the cropping of the acquired frames to obtain the individual images of the three samples, or stage 5 of the feature extraction process as explained in section 4.2, the method could be further automated using a Canny algorithm for border detection as explored by Yang et al [81]. This image processing technique for edge detection automatically finds the boundaries of objects in images by detecting discontinuities in brightness, and can be easily applied in MATLAB using the "edge" object [82]. A mask is applied to the image and the desired section can be saved. This method was explored by the author, however, the slight differences in brightness in the images did not allow easy identification of the borders. To achieve this, pre-processing of the images would be necessary which is a case for future work. Other future improvements and options for further studies include using the standard camera provided with the EOS M250 printing machine

to reduce experiment costs and standardize the feature extraction process, validating the software developed with a thermal video of the printing process, and utilising the software for videos of printing process of more complex geometries.

To summarize, the author developed a software for layerwise image collection and database creation for analysis of metal AM SLM printing processes, achieving a 94.5% and an adjusted 99% accuracy. The lack of studies regarding the matter and the generality of the process developed render the software a valuable tool for image extraction, especially for ex-situ and post-printing layerwise quality monitoring. Further improvements of the programme include the cross-validation of the frames saved based on other colour spaces and the automation of the image cropping using edge detection algorithms. Future studies to further validate the results and applicability of the software include the analysis of a video recorded by cameras incorporated to standard AM printers, the analysis of a video of a more complex geometry and the analysis of a video of thermal imaging of the printing process.

6.2. Analysis of Sample Classification Algorithm

Chapter 4.3 describes the methodology to create a supervised ML model for classification of layerwise images of the SLM printing process. Through the use of a BoVW, the author was able to train and export an algorithm that generates feature vectors of the images and uses them to classify the images into three different categories denoted as S1, S2 and S3. These categories are associated with the process parameters shown on Table 4.2. To the author's knowledge, although many studies explore ML algorithms for detection of defects during metal AM, few studies describe the method utilised to match layerwise images to the process parameters established for printing. The use of a well-established and relatively simple ML classification method that is the BoVW, provides a quick and efficient solution to identification of samples based on their layer-by-layer characteristics.

The model generated can be of high value for future analysis of process parameters on SLM AM print quality. The algorithm is able to quickly match the layerwise images of the printing process to the sample it belongs to, and hence identifying the process parameters that have been selected for the sample. As stated in section 3.2, to achieve in situ quality monitoring it is required to analyse the effect of process parameter on build quality, and real time monitoring involves high speed classification of said parts. Additionally, to obtain and accuracy of over 90%, most machine learning classifiers require more than one descriptor [83]. This translates in higher complexity and the prediction time. Nonetheless, the system presented in this work utilises only one descriptor and deliver over 90% accuracy, hence both speed and accuracy are achieved. This will be of great value for the creation and implementation of future monitoring systems. Lastly, as

stated above a clear advantage of the model is the flexibility and simplicity. In the case that newer categories and process parameters are implemented, the algorithm can be quickly trained, and the images do not require pre-processing to do so.

The parameters selected for the BoVW training play an important role in algorithm performance. The decision to employ the upright SURF descriptor based on the superiority in robustness and speed would have to be proved by comparing the performance of other descriptors such as SIFT, FAST or BRIEF; and while that is out of scope of this project, the author estimates an average 12% speed improvement based on past studies on the matter [26, 28]. As the video and frames are still and no rotation information has to be captured, the utilization of the scale invariant 'upright' feature descriptor not only increases speed of analysis but also discriminative power. Additionally, selecting the Grid method for selecting point locations for feature extraction improved the training accuracy results a 15% on average, independent of vocabulary size or other parameters changed.

The result-orientated iterative process of tuning the BoVW parameters proved workable and simple. As it can be observed from Table 5.1, the initial vocabulary size was set to 20, and was reduced until reaching a dictionary of 6 words which avoided model overfitting. The training and testing accuracy divergence shown by the first two iterations demonstrate how a larger vocabulary size than required results in overfitting. Notably, most selected training algorithms were either variations of KNNs or SVMs, which show their predominance over other algorithms. In the case of percentage of strongest features used to create the fingerprints of the images, a value of 80% was initially set and proved the most accurate even after several iterations, as reducing the value would result in underfitting and reduce algorithm accuracy and increasing the value augmented speed of training and testing and produced slight overfitting. In addition to that, the use of the classification learner in MATLAB proved invaluable as over 25 classification methods could be trained and compared. All in all, final tuning led to a 7-word vocabulary size, 80% of strongest features selected and the use of a Subspace KNN for algorithm training, achieving a 93.5% training accuracy and 92.1% testing accuracy.

The confusion matrix and scatter plot enable easy visualisation of algorithm performance and gaps for improvement. From the confusion matrix (Fig. 5.2), the exact number of misclassified images can be obtained. From that and the FNR, it can be observed that the most 'confused' images were from Sample 1 (13.3%), and the least form Sample 3 (6.20%). Notably, Sample 1 and 3 were not misclassified between each other, as shown by the white boxes in the top right and bottom left of the confusion matrix. Both these results are attributed to the comparison of feature vectors or fingerprints obtained from the BoVW, meaning that Sample 2 and Sample 1 are the most difficult

to define as two different clusters, seen in the scatter plot (Fig. 5.1), and Sample 1 and 3 have very different fingerprints on average. Additionally, Sample 2 constitutes 62.9% of all misclassified images. Overall, the algorithm obtained a testing accuracy of 92.1%, which is good given the image definition and quality, and a classification time for the testing set of 225 images in total of 10.7 seconds, a fast time given that each layer takes around 12 seconds to be completed. However, several modifications and improvements could be done to achieve a higher overall accuracy and speed.

A better positioning of the Xiris™ camera or the use of the standard camera of the EOS M250 printer may have rendered better results. While a high-quality video was recorded as specified on Table 4.1, the post-processed images obtained of the layer-by-layer samples did not provide enough detail for accurate distinction. From Fig. 4.5, it can be observed that a large area of the printing process is discarded after cropping the images.

A larger training and testing dataset can increase model accuracy and validity. In the end, only 567 frames were utilised, 270 for training, 69 for validation and 228 for testing; a third of each belonging to a sample. Recording a longer printing process or printing multiple geometries has the potential of highly increasing algorithm accuracy. However, as the number of training samples increases, the generation of histograms for visual words requires more computational power. Training the algorithm in the cloud is in that case recommended to avoid memory space availability problems.

The BoVW algorithm's computing speed could be enhanced even further by explicit parallelization of the layerwise analysis process, as explored by Scime et al [57]. This would allow for several MATLAB instances to be run on several processors, with separate memories, and execute a single function simultaneously.

To summarize, the supervised ML model created based on the BoVW classifier achieved high computation speeds and a 92.1% accuracy, which, in addition to the algorithm simplicity and flexibility, demonstrate the value and usefulness for future studies of the effect of processing parameter in metal AM part quality. Additionally, the validation techniques used greatly aid the visualisation of the results, helping the author identify weak points as well as demonstrate the robustness of the algorithm created. Moreover, several methods for improvement of the algorithm accuracy and validation have been raised, such as the relocation of the camera used, the gathering of a larger database and the use of parallel computing to increase the speed of the algorithm training and testing.

7. Conclusions

7.1. Summary

The study intended to meet two main objectives established at the start of the project:

- To automate the extraction of desired frames of a video of the AM process.
- To generate a supervised algorithm for the classification of the images in the database.

Having presented the results obtained and discussed their validity and implications, it can be stated that both criteria have been met. To fulfil both objectives, a video of the SLM printing process of three samples with distinct printing parameters was recorded using the Xiris™ camera.

Then, through the analysis of the change in mean b^* colour space values of all video frames, a software for identification and extraction of the desired features was created. A database of 567 frames divided in three sets of 189 cropped images of the surface of each layer of each of the three samples printed using an SLM machine was acquired. Although the recording of the printing contained all 200 layers of the samples, an error in camera focus settings resulted in the neglection of 9 of those layers. Even so, the software implemented did not capture any unwanted frames, hence proving its validity in unanticipated scenarios. For that reason, an adjusted accuracy of 99% was stated, having missed two of the 191 frames recorded. The main contributions that this work provides are:

1. The software will be able to identify required frames in most recorded videos of L-PBF printing processes since a change in mean b^* values will occur in all videos. This can be applied to any light spectrum.
2. The software is simple and intuitive, allowing further research to be carried out and modifications made in the case of changing parameters.
3. The software is invariant to printing cycles, hence it is able to acquire frames of complex and varying printing processes.
4. The software is ready to be employed in ex-situ and in-situ monitoring applications, and preliminary work can be done for real-time AM monitoring, although future work is suggested in section 7.2 for demonstrating said capability.

Subsequently, using the database acquired, an ML algorithm for the classification of the images of the printing process based on their visual information was developed. The author first selected the most appropriate model to carry out the classification task, a BoVW, due to its overall simplicity, adaptability and fast computing time. Then, using three fifths of all frames gathered, the model was trained, and a training accuracy of 93.5% was obtained. The training accuracy was

confirmed as valid, as 5-fold cross validation and an evaluation of the scatter plot obtained were performed. The two fifths of data images that were not utilised for algorithm training were employed for testing, obtaining a 92.1% data accuracy in 10.72 seconds of elapsed training time. A confusion matrix helped with the visualisation of the results obtained, and the obtention of the FNR allowed for analysis of common mistakes. It was observed that Sample 1 and 2 were confused between the most, the latter causing 62.9% of all total misclassifications. From the analysis, the author was able to identify main areas for improvement: a better positioning of the camera recording the video and a larger dataset of images for algorithm training. The author outlined contributions of the work presented to the AM community below:

1. The algorithm is able to classify layer-wise images at a high accuracy and low speed, which will allow quick analysis of effects of process parameters in each classified image.
2. The algorithm is simple and flexible, allowing quick training to be done in the case that more classifying labels are added.
3. The algorithm can be implemented into real-time quality monitoring and defect detection processes, easing the classification aspect of the task.

7.2. Recommendations for Further Work

As a whole, the study demonstrated the suitability of implementing software and ML models to speed up the obtention of image data from L-PBF printing and classification of them based on their visual feature vectors. These results suggest numerous available avenues to expand what was studied and perform future research, which may increase the likelihood of metal AM growth and inclusion in more industries. Below are these future suggestions:

1. Include other mean colour values into the if statement to act as cross-validations of frames acquired.
2. Explore Canny algorithms for border detection to automate image cropping and utilise for more complex geometries.
3. Explore co-axial CV methods to improve image quality and suitability, such as the standard cameras present in EOS M250s.
4. Establish relationship between frame feature vectors and histograms and process parameters to identify overarching patters.
5. Sample additional layer-wise surface images to increase datasets. With a larger dataset, the accuracies can be improved, and additional categories based on process parameters can be used.
6. Implement explicit parallelization to increase training and testing speeds.

References

[1] A. N. Olaf Diegel, Damien Motte, *A practical guide to design for additive manufacturing*, 1 ed. (Springer series in additive manufacturing). Springer Singapore, 2019, pp. XX, 226.

[2] P. Viola, M. J. Jones, and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance," *International Journal of Computer Vision,* vol. 63, no. 2, pp. 153-161, 2005/07/01 2005, doi: 10.1007/s11263-005-6644-8.

[3] S. A. M. Tofail, E. P. Koumoulos, A. Bandyopadhyay, S. Bose, L. O'Donoghue, and C. Charitidis, "Additive manufacturing: scientific and technological challenges, market uptake and opportunities," *Materials Today,* vol. 21, no. 1, pp. 22-37, 2018, doi: 10.1016/j.mattod.2017.07.001.

[4] T. Debroy *et al.*, "Additive manufacturing of metallic components – Process, structure and properties," *Progress in Materials Science,* vol. 92, pp. 112-224, 2018, doi: 10.1016/j.pmatsci.2017.10.001.

[5] D. Thomas and S. Gilbert, *Costs and cost effectiveness of additive manufacturing: A literature review and discussion*. 2015, pp. 1-96.

[6] E. Atzeni and A. Salmi, "Economics of additive manufacturing for end-usable metal parts," *The International Journal of Advanced Manufacturing Technology,* vol. 62, 10/01 2012, doi: 10.1007/s00170-011-3878-1.

[7] D. Herzog, V. Seyda, E. Wycisk, and C. Emmelmann, "Additive manufacturing of metals," *Acta Materialia,* vol. 117, pp. 371-392, 2016, doi: 10.1016/j.actamat.2016.07.019.

[8] S. Sun *et al.*, "Electron beam additive manufacturing of Inconel 718 alloy rods: Impact of build direction on microstructure and high-temperature tensile properties," *Additive Manufacturing,* vol. 23, 08/01 2018, doi: 10.1016/j.addma.2018.08.017.

[9] D. Ye, G. S. Hong, Y. Zhang, K. Zhu, and J. Y. H. Fuh, "Defect detection in selective laser melting technology by acoustic signals with deep belief networks," *The International Journal of Advanced Manufacturing Technology,* vol. 96, no. 5-8, pp. 2791-2801, 2018, doi: 10.1007/s00170-018-1728-0.

[10] B. Redwood, "Additive Manufacturing Technologies," D. Hubs, Ed., ed, 2016.

[11] J. H. Tan, W. L. E. Wong, and K. W. Dalgarno, "An overview of powder granulometry on feedstock and part performance in the selective laser melting process," *Additive Manufacturing,* vol. 18, pp. 228-255, 2017, doi: 10.1016/j.addma.2017.10.011.

[12] E. O. Olakanmi, R. F. Cochrane, and K. W. Dalgarno, "A review on selective laser sintering/melting (SLS/SLM) of aluminium alloy powders: Processing, microstructure, and properties," *Progress in Materials Science,* vol. 74, pp. 401-477, 2015/10/01/ 2015, doi: https://doi.org/10.1016/j.pmatsci.2015.03.002.

[13] G. Rebala, A. Ravi, and S. Churiwala, *An Introduction to Machine Learning*. 2019.

[14] H. Baumgartl, J. Tomas, R. Buettner, and M. Merkel, "A deep learning-based model for defect detection in laser-powder bed fusion using in-situ thermographic monitoring," *Progress in Additive Manufacturing,* vol. 5, no. 3, pp. 277-285, 2020, doi: 10.1007/s40964-019-00108-3.

[15] M. Kubat, *An Introduction to Machine Learning*. 2017.

[16] A. Ng, "Machine Learning," ed. Stanford University: Coursera, 2018.

[17] A. I. Khan and S. Al-Habsi, "Machine Learning in Computer Vision," *Procedia Computer Science,* vol. 167, pp. 1444-1451, 2020, doi: 10.1016/j.procs.2020.03.355.

[18] W.-M. Lee, *Python Machine Learning*. Indianapolis: John Wiley & Sons, 2019.

[19] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," 2005: IEEE, doi: 10.1109/iccv.2005.171. [Online]. Available: https://dx.doi.org/10.1109/iccv.2005.171,

[20] Y. V. Tiumentsev and M. V. Egorchev, "Dynamic Neural Networks: Structures and Training Methods," Elsevier, 2019, pp. 35-91.

[21] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *arXiv pre-print server,* 2014-11-06 2014, doi: None

arxiv:1411.1792.

[22] G. V. L. De Lima, P. T. M. Saito, F. M. Lopes, and P. H. Bugatti, "Classification of texture based on Bag-of-Visual-Words through complex networks," *Expert Systems with Applications,* vol. 133, pp. 215-224, 2019, doi: 10.1016/j.eswa.2019.05.021.

[23] L. Bampis and A. Gasteratos, "Revisiting the Bag-of-Visual-Words model: A hierarchical localization architecture for mobile systems," *Robotics and Autonomous Systems,* vol. 113, pp. 104-119, 2019, doi: 10.1016/j.robot.2019.01.004.

[24] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding,* vol. 110, no. 3, pp. 346-359, 2008, doi: 10.1016/j.cviu.2007.09.014.

[25] K. Zagoris, I. Pratikakis, A. Antonacopoulos, B. Gatos, and N. Papamarkos, "Distinction between handwritten and machine-printed text based on the bag of visual words model," *Pattern Recognition,* vol. 47, no. 3, pp. 1051-1062, 2014, doi: 10.1016/j.patcog.2013.09.005.

[26] M. Kashif, T. M. Deserno, D. Haak, and S. Jonas, "Feature description with SIFT, SURF, BRIEF, BRISK, or FREAK? A general question answered for bone age assessment," *Computers in Biology and Medicine,* vol. 68, pp. 67-75, 2016, doi: 10.1016/j.compbiomed.2015.11.006.

[27] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision,* vol. 60, p. 91, 11/01 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[28] E. Karami, S. Prasad, and M. Shehata, *Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images*. 2015.

[29] The MathWorks Inc. "bagOFFeatures" [Online] Available: https://es.mathworks.com/help/vision/ref/bagoffeatures.html

[30] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," Springer Berlin Heidelberg, 2010, pp. 778-792.

[31] J.-H. Wang, J. Ren, W. Liu, X.-Y. Wu, M.-X. Gao, and P.-K. Bai, "Effect of Selective Laser Melting Process Parameters on Microstructure and Properties of Co-Cr Alloy," *Materials,* vol. 11, no. 9, p. 1546, 2018, doi: 10.3390/ma11091546.

[32] T. G. Spears and S. A. Gold, "In-process sensing in selective laser melting (SLM) additive manufacturing," *Integrating Materials and Manufacturing Innovation,* vol. 5, no. 1, pp. 16-40, 2016, doi: 10.1186/s40192-016-0045-4.

[33] W. J. Sames, F. A. List, S. Pannala, R. R. Dehoff, and S. S. Babu, "The metallurgy and processing science of metal additive manufacturing," *International Materials Reviews,* vol. 61, no. 5, pp. 315-360, 2016, doi: 10.1080/09506608.2015.1116649.

[34] I. A. Okaro, S. Jayasinghe, C. Sutcliffe, K. Black, P. Paoletti, and P. L. Green, "Automatic fault detection for laser powder-bed fusion using semi-supervised machine learning," *Additive Manufacturing,* vol. 27, pp. 42-53, 2019, doi: 10.1016/j.addma.2019.01.006.

[35] R. K. Enneti, R. Morgan, and S. V. Atre, "Effect of process parameters on the Selective Laser Melting (SLM) of tungsten," *International Journal of Refractory Metals and Hard Materials,* vol. 71, pp. 315-319, 2018/02/01/ 2018, doi: https://doi.org/10.1016/j.ijrmhm.2017.11.035.

[36] M. Grasso and B. M. Colosimo, "Process defects andin situmonitoring methods in metal powder bed fusion: a review," *Measurement Science and Technology,* vol. 28, no. 4, p. 044005, 2017, doi: 10.1088/1361-6501/aa5c4f.

[37] S. K. Everton, M. Hirsch, P. Stravroulakis, R. K. Leach, and A. T. Clare, "Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing," *Materials & Design,* vol. 95, pp. 431-445, 2016, doi: 10.1016/j.matdes.2016.01.099.

[38] S. Berumen, F. Bechmann, S. Lindner, J.-P. Kruth, and T. Craeghs, "Quality control of laser- and powder bed-based Additive Manufacturing (AM) technologies," *Physics Procedia,* vol. 5, pp. 617-622, 2010, doi: 10.1016/j.phpro.2010.08.089.

[39] S. Clijsters, T. Craeghs, S. Buls, K. Kempen, and J. P. Kruth, "In situ quality control of the selective laser melting process using a high-speed, real-time melt pool monitoring system," (in English), *The International Journal of Advanced Manufacturing Technology,* vol. 75, no. 5-8, pp. 1089-1101, Nov 2014, doi: http://dx.doi.org/10.1007/s00170-014-6214-8.

[40] T. Craeghs, F. Bechmann, S. Berumen, and J.-P. Kruth, "Feedback control of Layerwise Laser Melting using optical sensors," *Physics Procedia,* vol. 5, pp. 505-514, 2010, doi: 10.1016/j.phpro.2010.08.078.

[41] M. Pavlov, M. Doubenskaia, and I. Smurov, "Pyrometric analysis of thermal processes in SLM technology," *Physics Procedia,* vol. 5, pp. 523-531, 2010, doi: 10.1016/j.phpro.2010.08.080.

[42] U. Thombansen, A. Gatej, and M. Pereira, "Process observation in fiber laser–based selective laser melting," *Optical Engineering,* vol. 54, no. 1, p. 011008, 2014, doi: 10.1117/1.oe.54.1.011008.

[43] J. A. Kanko, A. P. Sibley, and J. M. Fraser, "In situ morphology-based defect detection of selective laser melting through inline coherent imaging," *Journal of Materials Processing Technology,* vol. 231, pp. 488-500, 2016, doi: 10.1016/j.jmatprotec.2015.12.024.

[44] A. Neef, V. Seyda, D. Herzog, C. Emmelmann, M. Schönleber, and M. Kogel-Hollacher, "Low Coherence Interferometry in Selective Laser Melting," *Physics Procedia,* vol. 56, pp. 82-89, 2014, doi: 10.1016/j.phpro.2014.08.100.

[45] M. Abdelrahman, E. W. Reutzel, A. R. Nassar, and T. L. Starr, "Flaw detection in powder bed fusion using optical imaging," *Additive Manufacturing,* vol. 15, pp. 1-11, 2017, doi: 10.1016/j.addma.2017.02.001.

[46] B. Foster, E. Reutzel, A. Nassar, B. T. Hall, S. Brown, and C. Dickman, "Optical, layerwise monitoring of powder bed fusion," 2015.

[47] F. Imani, A. Gaikwad, M. Montazeri, P. Rao, H. Yang, and E. W. Reutzel, *Layerwise In-Process Quality Monitoring in Laser Powder Bed Fusion*. 2018.

[48] W. S. Land, B. Zhang, J. Ziegert, and A. Davies, "In-Situ Metrology System for Laser Powder Bed Fusion Additive Process," *Procedia Manufacturing,* vol. 1, pp. 393-403, 2015, doi: 10.1016/j.promfg.2015.09.047.

[49] Z. Li *et al.*, "In Situ 3D Monitoring of Geometric Signatures in the Powder-Bed-Fusion Additive Manufacturing Process via Vision Sensing Methods," *Sensors,* vol. 18, no. 4, p. 1180, 2018, doi: 10.3390/s18041180.

[50] B. Zhang, J. Ziegert, F. Farahi, and A. Davies, "In situ surface topography of laser powder bed fusion using fringe projection," *Additive Manufacturing,* vol. 12, pp. 100-107, 2016, doi: 10.1016/j.addma.2016.08.001.

[51] PricewaterhouseCooper, "3D Printing and the new shape of industrial manufactuting," p. 22, 2014.

[52] P. Mercelis, J.-P. Kruth, and J. Vaerenbergh, "Feedback control of Selective Laser Melting," *Proc. of the 15th Int. Symp. on Electromachining,* pp. 421-426, 01/01 2007.

[53] I. C. Ltd. (2020) Metal AM *The magazine for te metal additive manufacturing industry*. 168.

[54] P. Yadav, O. Rigo, C. Arvieu, E. Le Guen, and E. Lacoste, "In Situ Monitoring Systems of The SLM Process: On the Need to Develop Machine Learning Models for Data Processing," *Crystals,* vol. 10, no. 6, p. 524, 2020, doi: 10.3390/cryst10060524.

[55] Y. Chivel, "Optical In-Process Temperature Monitoring of Selective Laser Melting," *Physics Procedia,* vol. 41, pp. 904-910, 2013, doi: 10.1016/j.phpro.2013.03.165.

[56] S. Clijsters, T. Craeghs, S. Buls, K. Kempen, and J.-P. Kruth, "In situ quality control of the selective laser melting process using a high-speed, real-time melt pool monitoring system," *International Journal of Advanced Manufacturing Technology,* vol. 75, pp. 1089-1101, 08/10 2014, doi: 10.1007/s00170-014-6214-8.

[57] L. Scime and J. Beuth, "Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm," *Additive Manufacturing,* vol. 19, pp. 114-126, 2018, doi: 10.1016/j.addma.2017.11.009.

[58] S. S. Razvi, S. Feng, A. Narayanan, Y.-T. Lee, and P. Witherell, *A Review of Machine Learning Applications in Additive Manufacturing*. 2019.

[59] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, no. 6, pp. 84-90, 2017, doi: 10.1145/3065386.

[60] M. Aminzadeh and T. R. Kurfess, "Online quality inspection using Bayesian classification in powder-bed additive manufacturing from high-resolution visual camera images," *Journal of Intelligent Manufacturing,* vol. 30, no. 6, pp. 2505-2523, 2019, doi: 10.1007/s10845-018-1412-0.

[61] A. Caggiano, J. Zhang, V. Alfieri, F. Caiazzo, R. Gao, and R. Teti, "Machine learning-based image processing for on-line defect recognition in additive manufacturing," *CIRP Annals,* vol. 68, no. 1, pp. 451-454, 2019-01-01 2019, doi: 10.1016/j.cirp.2019.03.021.

[62] C. Gobert, E. W. Reutzel, J. Petrich, A. R. Nassar, and S. Phoha, "Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging," *Additive Manufacturing,* vol. 21, pp. 517-528, 2018, doi: 10.1016/j.addma.2018.04.005.

[63] M. Grasso, V. Laguzza, Q. Semeraro, and B. M. Colosimo, "In-Process Monitoring of Selective Laser Melting: Spatial Detection of Defects Via Image Data Analysis," *Journal of Manufacturing Science and Engineering,* vol. 139, no. 5, p. 051001, 2017, doi: 10.1115/1.4034715.

[64] J. Petrich, C. Gobert, S. Phoha, A. Nassar, and E. Reutzel, "Machine learning for defect detection for PBFAM using high resolution layerwise imaging coupled with post-build CT scans," 2017.

[65] Y. Zhang, G. S. Hong, D. Ye, K. Zhu, and J. Y. H. Fuh, "Extraction and evaluation of melt pool, plume and spatter information for powder-bed fusion AM process monitoring," *Materials & Design,* vol. 156, pp. 458-469, 2018, doi: 10.1016/j.matdes.2018.07.002.

[66] EOS. EOS M290 System Datasheet

[67] S. A. Shevchik, C. Kenel, C. Leinenbach, and K. Wasmer, "Acoustic emission for in situ quality monitoring in additive manufacturing using spectral convolutional neural networks," *Additive Manufacturing,* vol. 21, pp. 598-604, 2018, doi: 10.1016/j.addma.2017.11.012.

[68] A. Campilho and F. Karray, *Image Analysis and Recognition: 13th International Conference, ICIAR 2016, in Memory of Mohamed Kamel, Póvoa de Varzim, Portugal, July 13-15, 2016, Proceedings*. 2016.

[69] S. Abouzahir, M. Sadik, and E. Sabir, "Bag-of-visual-words-augmented Histogram of Oriented Gradients for efficient weed detection," *Biosystems Engineering,* vol. 202, pp. 179-194, 2021, doi: 10.1016/j.biosystemseng.2020.11.005.

[70] W. Shen, M. Zhou, F. Yang, C. Yang, and J. Tian, "Multi-scale Convolutional Neural Networks for Lung Nodule Classification," *Information processing in medical imaging : proceedings of the ... conference,* vol. 24, pp. 588-99, 2015.

[71] S. Pang and L. Xia, "Automatic classification and identification algorithms for single-environment risk and their application in social changes," *Technological Forecasting and Social Change,* vol. 166, p. 120642, 2021, doi: 10.1016/j.techfore.2021.120642.

[72] Xiris. Xiris XVC-1000/1100 Weld Camera [Online] Available: https://www.xiris.com/xiris-xvc-1000/

[73] K. O. D. GmbH. Kowa LM50JC10M C-Mount Lens [Online] Available: https://www.kowa-lenses.com/en/lm50jc10m-10mp-industrial-lens-c-mount?c=145

[74] The MathWorks Inc. "VideoReader" [Online] Available: https://es.mathworks.com/help/matlab/ref/videoreader.html

[75] The MathWorks Inc, ""rgb2lab"," 2017. [Online]. Available: https://es.mathworks.com/help/images/ref/rgb2lab.html.

[76] J. Schanda and I. Illumination, *Colorimetry: Understanding the CIE System*. 2007.

[77] The MathWorks Inc. "Min" [Online] Available: https://es.mathworks.com/help/matlab/ref/min.html?lang=en

[78] The MathWorks Inc. "imageDatastore" [Online] Available: https://es.mathworks.com/help/matlab/ref/matlab.io.datastore.imagedatastore.html

[79] The MathWorks Inc, ""Classification Learner"," 2016. [Online]. Available: https://es.mathworks.com/help/stats/classification-learner-app.html.

[80] D. B. Jinan Fiaidhi, N. Thirupathi Rao, *Smart Technologies in Data Science and Communication*, 1 ed. (Lecture Notes in Networks and Systems, no. 105). Springer Singapore, 2019, pp. XVIII, 318.

[81] Y. Yang, X. Zhao, M. Huang, X. Wang, and Q. Zhu, "Multispectral image based germination detection of potato by using supervised multiple threshold segmentation model and Canny edge detector," *Computers and Electronics in Agriculture,* vol. 182, p. 106041, 2021, doi: 10.1016/j.compag.2021.106041.

[82] The MathWorks Inc, ""edge"," 2020. [Online]. Available: https://es.mathworks.com/help/images/ref/edge.html.

[83] J. M. Dos Santos, E. S. De Moura, A. S. Da Silva, J. M. B. Cavalcanti, R. D. S. Torres, and M. L. A. Vidal, "A signature-based bag of visual words method for image indexing and search," *Pattern Recognition Letters,* vol. 65, pp. 1-7, 2015, doi: 10.1016/j.patrec.2015.06.023.

Appendix

Additional details and information are provided in the appendices.

The following link gives access to a repository of the code developed:

https://github.com/RomanVazquezL/Classification-AdditiveManufacturing.git

The following link gives access to a folder containing the training and testing datasets:

https://livewarwickac-my.sharepoint.com/:f:/g/personal/u1835236_live_warwick_ac_uk/EjNpvR_i-O1Bgafe3STI9LgBrJbsjx-cWtYc20jGpAWp8A?e=z9PDTe

Appendix A: Code for image database collection

Read video and initialise variables.

```
% Reading video
v = VideoReader('FullVideo.avi');
% Obtain number of frames in video
NumOfFrames = v.NumFrames;
% Set Up matrices
meanB = zeros(NumOfFrames, 1); % Matrix containing mean b* values
mean50B = zeros(50,1); % Matrix containing mean b* values of past 50 frames
minimum = zeros(NumOfFrames-50, 1); % Matrix containing minium values of past 50 frames
```

Identifying frames based on mean b* values.

```
% Initial for loop to obtain Mean b* of all frames
for frame = 1:NumOfFrames
    thisFrame = read(v,frame);
    cform = makecform('srgb2lab'); % Change frame from rgb to lab colour space
    LabFrame = applycform(im2double(thisFrame),cform);
    meanB(frame) = mean2(LabFrame(:,:,3)); % Obtain mean b* value
end

% For loop to obtain array of minimum values
for i = 1:(NumOfFrames - 50)
    mean50B = meanB(i:i+50);
    minimum(i) = min(mean50B); % Minimum value os past 50 frames mean b*
end

% For loop to write and compare mininum
for j = 7:(NumOfFrames-50)
    if minimum(j) < (minimum(j-6)/2) % Compare minimum values of current frame and 6th
frame prior
      outputBaseFileName = sprintf('Frame %4.4d.png', (j+44)); % File name based on
frame number
      outputFullFileName = fullfile('D:\', 'frames', outputBaseFileName); % Folder to
```

```
save frames in
      Frame = read(v,j+44); % Read frame
      imwrite(Frame, outputFullFileName); % Save frame in folder indicated above
    end
end
```

Delete files from same cycle that are unnecessary.

```
d = dir('D:\frames\Not valuable frames'); % Access folder
filenames = {d.name};% Access all filenames
Pnum = 00; % Define variable for previous file number

for i = 1:numel(filenames)
  fn = filenames{i}; % Access file name of specific file
  % Find all files that contain space and find first 2 digits of frame
  [num, cnt] = sscanf(fn(find(fn == ' ', 1, 'last')+1:end-6), '%d');
  % If statement to delete files with the same first 2 digits
  if cnt == 1 && isequal(num,Pnum)
    delete(fullfile('D:\frames\Not valuable frames', fn));
  end
  Pnum = num;
end
```

Manually select coordinates for image cropping

```
h = imshow('Frame 0431.png'); % Show first frame saved
hp = impixelinfo; % Show location of pointer in frame
set(hp,'Position',[5 1 300 20]); % Position of information of location
```

Save cropped images

```
d2 = dir('D:\frames') % Establis folder to analyse
filenames = {d2.name}; % Find all filenames in the folder

% For loop to save cropped images
for i = 3: numel(filenames) % For all files
    fn = filenames{iB; % Get current file number
    I = imread(fn); % Read frame of current file
    P1 = imcrop(I,[435 342 760 60]); % Crop frame for Sample 1 coordinates
    P2 = imcrop(I,[435 465 760 60]); % Crop frame for Sample 2 coordinates
    P3 = imcrop(I,[411 663 760 60]); % Crop frame for Sample 3 coordinates

    % Save cropped images to desired folder and with desired name, S1,S2,S3
    outputBaseFileName = sprintf('P1 %12s.png', fn);
    outputFullFileName = fullfile('D:\S1', outputBaseFileName);
    imwrite(P1, outputFullFileName);
    outputBaseFileName = sprintf('P2 %12s.png', fn);
    outputFullFileName = fullfile('D:\S2',outputBaseFileName);
    imwrite(P2, outputFullFileName);
    outputBaseFileName = sprintf('P3 %12s.png', fn);
    outputFullFileName = fullfile('D:\S3', outputBaseFileName);
```

```
    imwrite(P3, outputFullFileName);
end
```

Published with MATLAB® R2020b

Appendix B: Code for Image Classification

Establish datasets using ImageDataStore function.

```
File = fullfile('C:\Training and Test Set');
DB = imageDatastore(File,'IncludeSubfolders', true, 'LabelSource','foldernames');
```

Display class names and counts.

```
Table = countEachLabel(DB)
Categories = tbl.Label;
```

Partition 336 images for training and 228 for testing.

```
Ilocation = fileparts(DB.Files{1});
imgSet = imageSet(strcat(Ilocation,'\..'),'recursive'); % Recursively scan entire image
set folder
[training_set,test_set] = imgSet.partition(113); % Create training and test set
test_set = test_set.partition(76);
```

Create visual vocabulary.

```
tic % Start timer
% Use bagOfFeatures function to create vocabulary
% Set vocabulary size, type of feature point selection
% and amount of features to select from (Final was 7, grid, 80%)
BoVW = bagOfFeatures(training_set,
'VocabularySize',7,'PointSelection','Grid','StrongestFeatures', 0.8);
% Create array of presence of visual words on each image
imgdata = double(encode(BoVW, tr_set));
toc % Stop timer
return; % Stop running code
```

Visualise/Plot histograms of feature vectors.

```
% Plot histogram of random S1 image
img = read(training_set(1), randi(training_set(1).Count));
featureVector = encode(BoVW, img); % Encode data from BoVW
subplot(1,3,1); % Establish 1x3 image
bS1 = bar(featureVector, 'FaceColor', [0 0.7 0]);
titS1 = title({'Visual Word Occurrences of','Image in S1 Training Set'});
xlabS1 = xlabel('Visual Word Index');
```

```
ylabS1 = ylabel('Frequency ()');

% Set font and size
set(gca, 'FontName', 'Times New Roman')
set([xlabS1,ylabS1], 'FontSize', 13)
set([titS1], 'FontSize', 15)

% Plot histogram of random S2 image
img = read(tr_set(2), randi(tr_set(2).Count));
featureVector = encode(BoVW, img);
subplot(1,3,2);
bS2 = bar(featureVector, 'FaceColor', [0 0.5 0]);
titS2 = title({'Visual Word Occurrences of','Image in S2 Training Set'});
xlabS2 = xlabel('Visual Word Index');
ylabS2 = ylabel('Frequency (%)');

% Set font and size
set(gca, 'FontName', 'Times New Roman')
set([xlabS2,ylabS2], 'FontSize', 13)
set([titS2], 'FontSize', 15)

% Plot histogram of random S3 image
img = read(tr_set(3), randi(tr_set(3).Count));
featureVector = encode(BoVW, img);
subplot(1,3,3);
bS3 = bar(featureVector, 'FaceColor', [0 0.3 0]);
titS3 = title({'Visual Word Occurrences of','Image in S3 Training Set'});
xlabS3 = xlabel('Visual Word Index');
ylabS3 = ylabel('Frequency (%)');

% Set font and size
set(gca, 'FontName', 'Times New Roman')
set([xlabS3,ylabS3], 'FontSize', 13)
set([titS3], 'FontSize', 15)
```

Create a 339x8 table using the encoded features

```
ImageData = array2table(imgdata); % Convert array created earlier to table
% Link each category with each feature vector
Type = categorical(repelem({training_set.Description}', [training_set.Count], 1));
ImageData.Type = Type;
```

Use the image data to train a model and assess its performance using Classification learner

Open Classification Learner and import trained model

```
classificationLearner
```

Test out accuracy on testing set

```
tic % Start timer
testImData = double(encode(BoVW, test_set));
testImData = array2table(testImData,'VariableNames',M0780.RequiredVariables); %
Implement imported model
actualImType = categorical(repelem({test_set.Description}', [test_set.Count], 1)); %
Obtain actual category

predictedOutcome = M0780.predictFcn(testImData); % Implement imported model M0780 to
test

% Count correct predictions by comparing obtained classification and actual category
correctPredictions = (predictedOutcome == actualImType);
validationAccuracy = sum(correctPredictions)/length(predictedOutcome) % Calculate
validation accuracy
toc % Stop timer
```

Published with MATLAB® R2020b