FAULT DETECTION AND CLASSIFICATION IN AUTOMATED ASSEMBLY MACHINES USING MACHINE VISION

by

VEDANG DILIPKUMAR CHAUHAN

A thesis submitted to the Graduate Program in Mechanical and Materials Engineering

in conformity with the requirements for the degree of

Doctor of Philosophy

Queen's University

Kingston, Ontario, Canada

(April, 2016)

Abstract

Automated assembly machines operate continuously to achieve high production rates. Continuous operation increases the potential for faults, with subsequent machine downtime. Early fault detection can reduce the amount of downtime. Traditional fault detection methods check for deviations from fixed threshold limits with multiple mechanical, optical and proximity sensors. The goal of this thesis was to develop and validate a machine vision inspection (MVI) system to detect and classify multiple faults using a single camera as a sensor.

An industrial automated O-ring assembly machine that places O-rings on to continuously moving plastic carriers at a rate of over 100 assemblies per minute was modified to serve as the test apparatus. An industrial CCD camera with LED panel lights for illumination was used to acquire videos of the machine's operation. A Programmable Logic Controller (PLC) with a Human-Machine Interface (HMI) allowed for the generation of faults in a controlled fashion. Three MVI methods, based on from computer vision techniques available in the literature, were developed for this application. The methods used features extracted from the videos to classify the machine's condition. The *first method* was based on Gaussian Mixture Models (GMMs), as originally used for real-time outdoor tracking of moving regions in image sequences. The *second method* used an optical flow approach which was originally used for motion estimation in a video. The *third method* was based on running average and morphological image processing, originally used for noise filtering in image sequences.

In order to provide a single metric to quantify relative performance, a Machine Vision Performance Index (MVPI) was developed with five measures of performance: accuracy, processing time, speed of response, robustness against noise, and ease of implementation. On the basis of the calculated MVPI, it was concluded that the GMM-based method is the best of the three methods for this application. This thesis has two main contributions: 1) validation that MVI can be used to detect and classify multiple faults using a single camera and 2) documentation on how computer vision techniques can be applied to the problem of fault detection and classification in assembly machines.

Acknowledgements

Firstly, I would like express my sincere gratitude to my supervisor Dr. Brian W. Surgenor for his supervision of this project. He has provided to me an opportunity to learn both technical and professional aspects of engineering. I am very thankful to him for his encouragement, support and constant feedback throughout the course of my studies at Queen's University. He is a great teacher, supervisor and a nice person. Thank you Dr. Surgenor.

I would also like to thank Dr. Gene Zak and Dr. Michael Greenspan, members of my supervisor committee, for their suggestions and feedback. I am thankful to Dr.Ronald Anderson, Dr. Qingguo Li, Dr. Kevin Deluzio (Head of the department), Ms. Gabrielle Whan, Ms. Jane Davies, Mr. Onno Oosten, Mr. Chris Barker, Ms. Tammy Wintlet and administrative and technical staff of the Mechanical and Materials Engineering Department for all their help and support.

I must express my gratitude to Chaitali, my wife, for her continued support and encouragement during all these years. I am very happy to thank my son Aarya for his innocent mischiefs and his unconditional love. I would like to thank my parents, sister, in-laws and other family members for their love, support, advice and blessings in my life. From the bottom of my heart, I want to thank God, for being there for me, never leaving me, and always loving me!

I also want to recognize the financial support provided by the Mechanical and Materials Engineering Department, Queen's University, OSAP, R. S. McLaughlin Fellowship and the Queen Elizabeth II Graduate Scholarships in Science and Technology which has been crucial to my ability to continue my education.

I am fortunate to have the support and guidance from my friends, Amit, Kalpeshhai, Ketul, Ashishbhia, Ajay, Ronak, Keyur, Manunath, Arjun and their families throughout my study. I would also like to thank all my colleagues in the Intelligent Automation Lab: Heshan, Luis, Carlos A., Carlos G, Luciana, Vahid and Saad. Thank you all for your helpful ideas and friendship during the last three years.

Table of Contents

List of Figures

List of Tables

Nomenclature

Variables

F	Foreground frame
I	Image frame
I_x, I_y and I_t	Spatiotemporal image brightness derivatives with respect to x, y and t
X	History of pixels gray value
$backfrm, B$	Background frame
p_1	Measure of accuracy in fault detection
p_2	Number of frames processed in 10s
p_3	Measure of speed of response
p_4	Measure of robustness against noise
p_5	Measure of ease of implementation/tuning
MaxFlowDensity	Maximum optical flow from the set of training frames
mxarea	Maximum foreground area from the set of training frames
nf	Total number of frames
ntf	Number of training frames
nz	Number of frames with zero foreground area
OpfDensity	Optical flow density vector
OptFlow	Optical flow vector
RunAvgArea	Running average area vector
SeClose	Size of structuring element for morphological closing operation
SeOpen	Size of structuring element for morphological opening operation
T	O-ring thickness
Th	Threshold for maximum running average area from the set of training frames
v	Optical flow in y-direction

$Area, A$	Foreground object area vector in a frame
E	Global energy of a frame
$I(x, y, t)$	Image frame at time t
$MVPI, P$	Machine vision performance index
$P(x)$	Joint probability distribution
s	Structuring element
u	Optical flow in x-direction

Greek Variables

$\nabla^2 u$	Laplacian of u
$\nabla^2 v$	Laplacian of v
$\sum_c$	Cluster Covariance
μ_c	Cluster mean
π_c	Cluster size
ϕ_p	O-ring diameter
σ_c	Cluster variance
$\omega_{i,t}$	Estimate of the weight for ith Gaussian component
α	First learning rate
η	Gaussian probability density function
ρ	Second learning rate

Acronyms

AirNO	Air knife Normal Operation
ANNs	Artificial Neural Networks
ART	Adaptive Resonance Theory
Bp	Backpropagation
CCD	Charge-Coupled Device
CM	Confusion Matrix
CNC	Computerized Numerical Control
EM	Expectation Maximization
FN	False Negative
FOV	Field of View
FP	False Positive
FPR	False Positive Rate
GA	Genetic Algorithm
GMMs	Gaussian Mixture Models
GUI	Graphical User Interface
HMI	Human Machine Interface
HopperNO	Hopper Normal Operation
MVI	Machine Vision Inspection
MVPI	Machine Vision Performance Index
NO	Transfer Track Normal Operation
OFD	Optical Flow Density
PLC	Programmable Logic Controller
ROC	Receiver Operating Characteristics

ROI	Region of Interest
SDK	Software Development Kit
STV	Spatiotemporal Volume
T1JAM	Transfer Track 1 Jam
T2JAM	Transfer Track 2 Jam
TN	True Negative
TP	True Positive
TPR	True Positive Rate
WD	Working Distance

Chapter 1

Introduction

In the early days of manufacturing, the process of assembling individual parts into a finished product was performed manually by highly skilled craftsmen sitting at a workstation. In 1913, Henry Ford introduced the concept of an assembly line, where parts were added to a product in a sequential manner. This process was significantly faster than the handcrafted approach. With the advent of cheap and reliable microcontrollers, sensors and actuators, the assembly line became automated in the latter half of the 20th century. Automation was carried out in three different stages: 1) automation in the manufacturing of individual parts by using numerical controlled machines, 2) transfer of partly or completely finished parts on the shop floor using automatic transfer methods and 3) using robots and other programmable machines for automatic assembly of parts in the final finished products (Groover, 2001).

The use of machines for automated assembly enabled modern manufacturing industries to achieve high production throughput and to gain competitive advantages. To achieve these goals, automated machines are operated around-the-clock. Continuous operation of an assembly machine results in wear of its various mechanisms that in turn lead to machine faults such as part jams, missing parts in the assembly, misalignment, and blockages with subsequent machine downtime. The downtime is a serious concern because it reduces the production rate and consequently increases the cost of manufacturing. When machine failure occurs, a small portion of downtime is spent for actual repairing, while the majority of time is consumed in locating the source of the problem (Holloway et al., 1990). Even with the state of the art technology, it is hard to prevent the machines from having faults such as missing parts in assemblies, part jams, misalignments, and blockages. Early fault detection and classification can minimize production downtime and help to restore a machine to its online state in the least possible time (Boothroyd, 2005). Many researchers, from different fields, have studied and developed various fault detection and classification methods with different applications. Therefore, it is important to define some terminology for fault detection, classification and diagnosis that will be used in this dissertation. Detection is the first stage where a fault is recognized as soon as it occurs. Classification is the second stage that deals with detecting the type and location of the fault. Further information about the fault is obtained in the third stage where diagnosis determines the causes of the fault. This thesis is concerned with the development of a fault detection and classification system as applied to an automated O-ring assembly machine.

1.1 Problem Overview

Development of a fault detection and classification system for automated assembly machines is the subject of this research. A fault detection system has five main elements. The first element is a sensor that is used to monitor and acquire data from the machine. The type and location of the sensor is very important as it determines which faults can be detected and how fast they can be detected. Further, the sensor should be non-intrusive for the machine. The second element is a data acquisition system that configures and records the sensor's output in a suitable form. The third element is a feature selection technique. Features must be descriptive and robust. The fourth element is a classification method that uses features as inputs and classifies the operating condition into the category of fault or fault free. The fifth element is a communication channel to take the necessary actions once a fault is detected. PLCs (Programmable Logic Controllers) are often used for communication between machines and fault detection systems.

Traditional methods of fault detection in machines are based on readings from sensors (such as limit switches, proximity, potentiometers, pressure sensors, and current and voltage sensors) and monitoring them with limit checking, where a fault is detected when a sensor reading exceeds either upper or lower threshold limits. More recent methods are model-based methods, where machine input and output signals are used to generate a mathematical model of a process to determine the occurrence of faults. These methods are subjected to limitations in terms of added cost due to the need for multiple sensors and detecting a fault only after it exceeds the threshold, which might take a long time. Furthermore, it may not provide proper information for fault diagnosis. In high speed automated assembly machines, these traditional methods are often not fast enough for rapid and accurate fault detection and classification. Therefore, a faster automatic fault detection and classification method is needed to minimize the delay time in taking corrective action, once a fault is detected. Machine Vision Inspection (MVI) systems are popular for finished product inspection applications, robot guidance, object tracing, etc. These systems use industrial grade CCD (charge-coupled device) cameras to acquire data in the form of images and videos. A camera can be used for continuous video acquisition of the machine's operation and computer vision techniques could be used to develop a fault detection system that is non-intrusive, requires less processing time and can adapt to changing operating conditions. Advanced compact MVI systems use smart cameras that can acquire images, process them using a built-in processor and communicate a decision to the real world using its digital I/O lines. In the context of this research, a PC-based MVI system for fault detection and classification on an automated assembly machine is to be designed and developed.

1.2 Objectives

There are three main objectives for this thesis work. The first objective is to modify an existing O-ring assembly machine into a testbed that can be used for fault detection and classification using a MVI system. The machine was originally designed and developed as a prototype by a Kingston based automated assembly machine builder. The second objective is to design and develop a MVI system using computer vision/image processing algorithms that can detect and classify commonly observed faults on the machine. This objective leads to an investigation of the practicality of using single camera-based MVI methods to detect and classify faults. The third objective is to come up with a performance index that can be used for performance evaluation of different MVI algorithms. This will allow one to determine the best MVI based fault detection and classification method for the O-ring assembly machine. The following tasks are formalized to achieve these objectives.

- Modify the existing O-ring assembly machine: The O-ring assembly machine was originally designed to be the first stage of a larger machine where O-rings were to be transferred to the second stage of the machine and ultimately used in the assembly of coaxial cables. Thus, the O-ring machine needs to be modified to allow for the continuous recirculation of O-rings so it can be operated on a standalone basis.
- Design a controllable fault generation system: During the running of the machine for an extended period of time, certain faults are commonly observed. The occurrences of these faults is not consistent in terms of timing and location. Therefore, a system is needed that can be used to generate faults in a controlled fashion to obtain fault data for the MVI system.
- MVI system hardware selection: Camera and lighting are the two main hardware elements of a MVI system. The first step is selection of suitable camera with appropriate resolution, acquisition rate, sensor size, and data transmission protocol. The lighting plays an important role in data acquisition with the camera. High grade industrial lights can deliver the best results but they add significant cost to the system. Therefore, LED lights are to be used that can meet the minimum lighting requirements.
- Data collection for multiple operating conditions: Controlled faults are to be introduced using a PLC. Data are to be acquired for all operating conditions including normal and faulty conditions. For comparative purposes, all MVI methods should be tested on the same data, therefore, processing must be done off-line; however, processing times are considered as a benchmark.
- Develop and test MVI based fault detection and classification methods: A literature review of the related work will help to understand which methods have the potential to be used for fault detection

purposes. Using this information, new MVI methods will be developed and tested for fault detection on the O-ring assembly machine.

- Determine criteria for performance evaluation for each MVI method and compare their performance for the given application: In this step, at first, commonly used performance measurement criteria will be studied and then a performance index will be developed to compare performance of different MVI algorithms applied to the same data. Parameters such as accuracy, speed of fault detection, processing time, robustness against noise and ease of implementation will be considered as performance measures.

1.3 Thesis Outline

Chapter 2 provides background information and a literature review of topics related to fault detection and classification on automated assembly machines. The chapter begins with a brief description of fundamentals of automation, automated assembly machines and faults and the need for a better fault detection and classification system. The next section of the chapter covers the related work in fault detection and classification by other researchers. The research papers are included from both non-vision and vision related studies for fault detection and classification purposes. The last section of the chapter explains commonly used performance evaluation criteria for MVI methods.

Chapter 3 describes the experimental apparatus and MVI system hardware used for acquiring video data from the O-ring assembly machine. The first section of the chapter explains construction and operation of the O-ring assembly machine and modifications performed on the machine to use it as the test apparatus. It explains in detail the normal operation for the assembly process and the role of the pneumatic actuation system. During several runs of the machine for an extended period of time, a set of commonly occuring faults were observed. The next section of the chapter provides a description on type and location of the faults, their symptoms and causes, and a mechanism to introduce the faults in a controlled manner using the HMI (Human-machine Interface) and PLC. The last section covers the design and selection of supporting elements, namely, camera, lights, lens and working distance.

Chapter 4 explains the methodology followed for the development of three MVI methods for fault detection and classification on the O-ring machine. The first section provides an overview of experimental strategy considered for the fault detection problem. The next section presents a procedure for acquiring video data for various operating conditions, including both normal and faulty operations. The video data were acquired using two different camera setups. The reason for the two setups is given and the data acquisition conditions and criteria are discussed. These video files are used to detect and classify fault conditions using three MVI methods: (1) fault detection using GMMs (Gaussian Mixture Models) and blob

analysis, (2) fault detection using optical flow estimation and (3) fault detection using foreground running average area. The next three sections of the chapter explain the algorithm development for each of the three methods.

Chapter 5 presents the results of training and testing for each of the three MVI methods. The first section of the chapter provides the fault detection results with Method 1. Similarly, the second section of the chapter explains the results with Method 2 and the third section presents the results with Method 3. The next section of the chapter provides the performance measurement parameters calculated for all methods as applied to all video data sets. A comparison between the three methods in terms of the accuracy and speed of fault detection is given. The overall performance for each MVI method is calculated by the weighted sum of the individual performance measurements. The significance of the results is discussed in the last section of the chapter.

Chapter 6 summarizes the accomplished work and provides conclusions on the effectiveness of a MVI system for fault detection and classification. It summarizes which method worked best and which method performed poorly and explains the underlying reasons. Recommendations for future work to detect unknown faults and multiple simultaneous faults are presented in the last section of the chapter.

Chapter 2

Background and Literature Review

Fault detection and classification using a Machine Vision Inspection (MVI) system for an automated assembly machine is the subject of this thesis. Many researchers have worked in the area of fault detection and classification in machines. In the case of assembly machines, faults cause either machine failure or machine malfunction. Machine failure means permanent damage to the system while machine malfunction is the case where a machine is not able to perform its required function due to faults. Fault detection and fault classification are the two main elements of a typical fault monitoring system. Fault detection is the stage where a fault is recognized as soon as it occurs and fault classification means detecting type, location and other signature parameters of a particular fault. Most common methods of fault detection are traditional sensor based methods, whereas a network of sensors monitors the machine's operating condition by continuously reading the sensors output and comparing it with the desired value. Several papers were found for fault detection and classification using Petri nets, model-based methods, neural networks and fuzzy inference methods. MVI methods are commonly used for industrial applications such as part quality checks, robot guidance and liquid level monitoring and non-industrial applications such 3D mapping, surveillance, automated harvesting and crop grading. Using a MVI system for fault detection and classification is a relatively new concept and a few publications have been found in this category. Therefore, a literature review, summarized in this chapter, includes both non-vision and vision-based methods developed for inspection and classification purposes. The chapter begins by describing important elements of assembly automation and automated assembly machine faults in Section 2.1. Machine fault detection and classification methods are covered in Section 2.2. Literature related to MVI systems for automatic inspection tasks is reviewed in Section 2.3. Next to last, Section 2.4 lists the commonly used performance evaluation criteria for MVI systems. Finally, Section 2.5 summarizes the chapter.

2.1 Industrial Automation and Need for Fault Detection and Classification

Industrial automation can be defined as a technology concerned with the application of mechanical, electronic, and computer-based systems to operate and control production of goods (Groover, 2001). The automation era began in early 19th century with the development of controllers, which were able to perform necessary calculation and respond to deviations from a set point. Numerical machine tools were developed in 1950s for control of machines using a punched paper tapes and later, they were converted into Computerized Numerical Control (CNC). At that time the purpose of automation was to increase productivity (since automated machines can be operated continuously), and reduce the cost associated with

human operators. However, further advancement of technology has produced more powerful sensors, actuators, control-communication systems and HMIs that has shifted the focus of automation to increasing quality and flexibility in a manufacturing process. For example, the piston assembly task that performed manually with an error rate of 1%, can be now performed using automation with an error rate of 0.00001% (Sure Controls Inc., 2016).

Principal advantages of industrial automation are high productivity, high product quality with uniformity, greater flexibility of changing tasks using programming and minimum hardware changes, high information accuracy in terms of data collection and high safety while working in hazardous conditions. Due to less demand for a human operator in automated processing, there was a strong need for automatic supervision of machines and processes. For example, in a manual assembly process, a human operator can perform both the assembly and supervision/inspection of the assembly at the same time. Any faults in the assembly process can be directly addressed by the human operator. But in the case of automated assembly machines, a controller controls the machine and performs the assembly operations. The faults in the automated assembly machine, if they stay undetected, can damage the machine or the product. It can lead to machine downtime that can have a direct impact on the cost and productivity. Therefore, an automated inspection system is required to ensure reliable operation of automated machines (Szkilnyk, 2012).

2.1.1 Fundamentals of Automated Assembly Systems

An automated assembly system performs a sequence of assembly operations using mechanized and automated devices to combine multiple components into a single entity in an assembly line or cell. The single entity can be a final product or a subassembly in a larger product. Automated assembly system may involve a single or multiple work stations. Depending upon the physical configuration, an automated assembly system can be classified into four main configurations: (a) In-line assembly machine, (b) dial-type assembly machine, (c) carousel assembly system and (d) single station assembly machine. The work transfer systems in an automated assembly can be continuous, synchronous or asynchronous. In continuous transfer systems, part holders are always in motion while the assembly operations are being performed. In synchronous transfer system, all part holders either move or stop at the same time at different work stations. Asynchronous transfer systems have a continuous moving conveyor between different work stations and part holders have independent motion with respect to each other. The holders are taken off the conveyor at each assembly station and after the operation is over, they are released back to the conveyor.

The in-line assembly machine is the assembly version of the machining transfer line, where number of workstations are located along the transfer system and the base part is being transferred from station to station using synchronous and/or asynchronous transfer system. An automobile assembly line is an example

of in-line assembly machine. In the dial-type assembly machine, workstations are arranged in a circular fashion around the periphery of the dial. The base part is fixed at one of the positions on the dial and other components are added at the various workstations as the dial indexes. The cycle time consists of the operation time and the transfer time. Because of the part jams and other malfunctions, the machine operates at less than 100% uptime. In carousel automated assembly system, the workstations are located along the sides of the carousel and the part holders move with the carousel. The base part enters at one end of the carousel, components are added to it at each workstation and the assembly is released at the other end of the carousel. The system can use both synchronous and asynchronous transfer systems. The single station system uses only one workstation and a means of feeding and assembling individual assembly parts to the workstation. The base part is held in a position and the required components are added to it at the same station. However, the system is slower compared to previous automated assembly systems. The robotic assembly of printed circuit boards is an example of single station system. The assembly machine used in this thesis is a dial-type assembly machine, a single workstation with a part delivery system.

At each workstation in all of the above configurations, two main tasks are performed: (1) a part is delivered and added to the base part and (2) an assembly operation, such as fastening or joining, is carried out at the station to join the parts together. Therefore, a part delivery system is very important in assembly machine. It consists of five elements, namely, *hopper, part feeder, selector, feed track* and *escapement*. Details of the operation of the machine are given in Chapter 3. Figure 2-1 is provided to illustrate the terms in the context of the machine used for this thesis. A *hopper* is the container into which the parts are stored in bulk to supply them to the machine. A hopper usually contains a mechanism that removes parts from the hopper one at a time and delivers them to the workstation. A vibrating mechanism that shakes and moves the parts out from the hopper for the assembly process is an example of a *part feeder*. Most often hopper and part feeder are combined as a single unit such as a rotary bowl feeder. A *selector* and orientor are devices used to permit only the right orientation of parts and to reorient improperly oriented parts. The hopper and part feeder are located at a distance from the assembly workhead where parts are assembled. The *feed track* is used to transfer the parts from the hopper to the assembly workhead while maintaining the orientation of the parts. Two types of feed tracks are commonly used: a) gravity feed track where the hopper is located at an elevation from the assembly workhead and parts are delivered by means of gravity and b) powered feed track where parts are delivered using air pressure, vibration or by other means toward the assembly workhead. An *escapement* device removes parts at the final stage of part feeding and controls the feed rate of the parts to the assembly workhead. The placement device physically removes the part from the feeder and places it in the correct orientation for the assembly operation. Faults such as jams and incorrect orientation of parts may result if the escapement device is not designed properly.

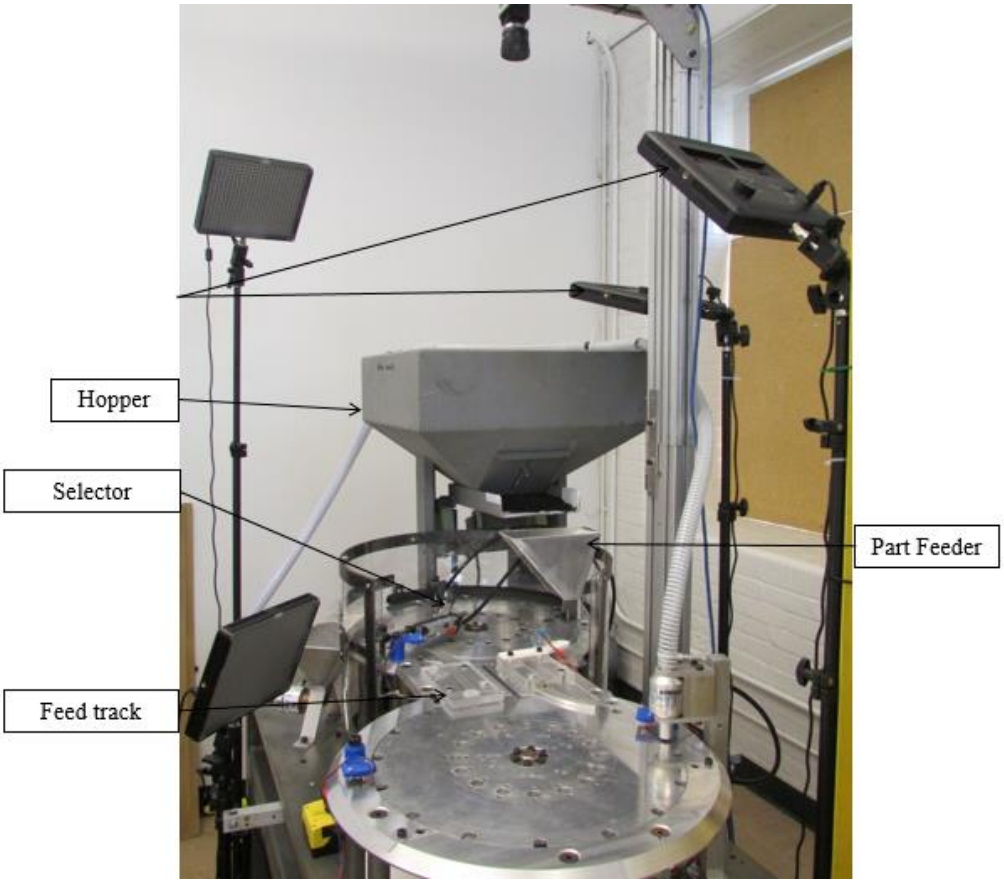


Figure 2-1 Automated O-ring assembly machine part delivery system

2.1.2 Automated Assembly Machines and Faults

Automation has increased the production rate with the help of high speed machines and transfer systems. With increases in production rate, the chances of equipment downtime are also increased. Automated machines are operated continuously, and that results in high friction and wear of its components. The result of wear, tear and friction for a longtime lead to either machine malfunction or failure. A malfunction is a temporary inability of a machine to perform its desired operation while the failure is a permanent interruption to the operation of the machine. Both cases require the machine to stop due to a fault and undergo the maintenance and repair work. One of the major concerns for manufacturing industries is the production downtime due to machine fault. "A fault is an unpermitted deviation of at least one characteristic property (feature) of the system from the acceptable, usual, standard condition" (Isermann, 2006). For this research, a machine fault is defined as a condition in a machine, where, as a result, the machine cannot perform its required task completely or partially. For

assembly systems, definitions of the types of faults were defined by (Szkilnyk, 2012). Table 2-1 summarizes the classification of automated assembly machine faults as originally defined by Szkilnyk.

There are two main categories of faults in assembly systems: (a) hard faults (b) soft faults. Hard faults are more serious and they demand immediate attention from an operator to stop a machine for corrective action. They prevent the machine from operating under its normal operation cycle. Soft faults do not require immediate action and the machine can continue operating up to certain time. If soft faults are kept unattended for a long time, they can convert into hard faults. Hard and soft faults can be further categorized as *equipment faults, software faults, task faults* and *product faults*. *Hard Equipment faults* are related to physical elements of the machine such as a broken gear or belt drive. Unbalanced rotating parts and leaks in pneumatic systems are considered soft equipment faults. *Software faults* include faults due to logical errors in algorithms or software bugs. If a software fault does not immediately affect the ability of a machine to operate then it is considered a soft software fault, otherwise it is a hard software fault. *Task faults* refer to faults in the operation sequence of assembly machines. Part jams and dropped parts from a gripper are hard task faults, while loosely held parts and stick slip motion of parts in tracks are soft task faults. *Product faults* are related to the quality of a product. Missing parts in an assembly and out of the specification product are examples of hard product faults. Soft product faults include the case of low quality product or a product that is difficult to use. The faults studied in this thesis are equipment, task and product faults, all of the hard faults category.

2.2 Related Work in Fault Detection and Classification

Early fault detection and classification can minimize production downtime and help to restore a machine to its online state in the least possible time (Boothroyd, 2005). A number of researchers have worked in the general area of fault detection and diagnosis in machines. During the literature review, thirty papers were found on the topic of machine condition monitoring and fault detection in assembly automation. None of the papers involved machine vision. A selection of the non-machine vision papers will be reviewed in this section.

Fault detection can be carried out by defining rules for normal operation of a machine and checking whether the rules are followed. A Petri net-based method was developed by Viswanadham and Johnson (1988) for fault detection and diagnosis of an automated manufacturing system using a two-level scheme. At Level 1, an intelligent controller performed the monitoring for each subsystem, such as machining centers, robots and conveyors. Level 2 had a Petri net based controller that kept track of work piece flow and communicated with the Level 1 controller for system level monitoring. The normal operation of the system was modeled using Petri nets, by collecting data from sensors and I/Os using PLCs. The detection

Table 2-1 Types of assembly machine faults (updated from Szkilnyk, 2012)

Failure/Faults in Assembly Systems								
Hard Faults					Soft Faults			
Equipment Faults	Software Faults	Task Faults	Product Faults		Equipment Faults	Software Faults	Task Faults	Product Faults
Examples	*Examples*	*Examples*	*Examples*		*Examples*	*Examples*	*Examples*	*Examples*
-Broken drive belt -Broken shaft -Seized bearing -Faulty sensor -Seized actuator	-Failed controller -Logical error in algorithm -Software bug	- Part improperly located on a fixture - Part jam in a track -Dropped part from a gripper -Improper assembly process	-Out of the specification product -Missing part in an assembly -Damaged or defective product		-Unbalanced rotating parts in the machine -Leaks in hydraulic and pneumatic systems -Sensor output out of the range readings -Excessive friction affects the motion of the parts	-Program outputs machine status signal to a wrong port -Extended code execution time	- Loosely held parts in a gripper -Stick slip motion of parts in a track - Slightly off the location part	- Little tight interference fit in the assembly -Product does not fully meet ergonomics standards -Product quality is not the best but acceptable

unit collected real time operating data, compared them with the data generated by the Petri net model and subsequently identified any deviations as faults. An approach based on a behavioral model of a discrete manufacturing system for automated fault detection and diagnosis was presented by Holloway et al. (1990). The approach was implemented by modeling the behavior of the system using signal flow from actuators to sensors though functional components. The approach was demonstrated with an example of a transfer mechanism having two conveyors, a signal actuator, a lift mechanism and three pulse type sensors. A base model was built first and then compared with the real-time model for detection of any deviations.

Konrad (1996) presented a fault detection method for milling machines using parameter estimation where a pattern classifier was built based on model parameters estimated for each insert of a milling cutter for a given cutting force. A pattern classifier was then used to detect and classify faults such as wear or breakage of inserts. Researchers have also used signal processing methods for fault detection purposes. For example, a time-frequency analysis method based on wavelet transform can detect faults from signals during the early stages of their occurrence and provide better information for the fault diagnosis. Analysis of vibration signals from a helicopter gearbox using wavelet transform was implemented for fault detection and diagnosis by Wang and McFadden (1996). Fault detection in automatic packaging machine using signal processing methods such as amplitude probability density and power spectral density was studied by Dalpiaz and Rivola (1997). The vibration signals for normal operation in a rotating machine were considered as the reference and real time vibration signals from the machine were recorded and compared with the reference model for fault detection and predictive maintenance.

Decision trees and artificial neural networks are widely used for fault detection and diagnosis in assembly automation. The ROBODOC, a decision tree based generalized system for fault diagnosis and maintenance of automated systems, was proposed by Patel et al. (1995). The approach used DCLASS, a group technology method for decision making and classification of faults and symptoms. It used both forward chaining and backward chaining inference mechanisms for identifying a fault. The proposed knowledge based expert system proved to be more efficient and faster than human experts and it was also used by non-experts for correcting faults without the help of maintenance specialists. Demetgul et al. (2009) implemented two Artificial Neural Networks (ANNs): Adaptive Resonance Theory (ART) and Backpropagation (Bp), for fault detection in a pneumatic modular production system. Both ANNs were trained using the data collected from eight different sensors for both normal and faulty operations and were able to correctly classify the faults using the trained networks. A Genetic Algorithm (GA) is often utilized for optimization of ANNs. A GA-ANNs combination was developed by Demetgul et al. (2011a) for fault detection in a bottle filling plant. The GA was used to design an optimum ANN structure with a minimum number of hidden layers, hidden nodes, mean-square error and response time. The data collected from three

pressure sensors and three position sensors were utilized for successfully classifying the plant operation into one normal and six fault conditions. Another study by Demetgul et al. (2011b) compared fault detection based on four types of classifiers: Bp, ART, fuzzy ARTMAP and fuzzy ART. A two axis servo-pneumatic system was used as the test set up. The results indicated that unsupervised ANNs such as ART and fuzzy ART performed equally well as that of supervised ANNs such as Bp and fuzzy ARTMAP. However, unsupervised ANNs were likely preferred for industrial applications as they do not require training. Fernando (2014) developed a system using three grayscale sensors and two limit switches for fault detection and identification in automated assembly machines. Two ANNs (Bp and ART) and the rule-based method were tested with data collected for normal and faulty operation sequence of the machine. Result showed that all three methods were able to achieve perfect classification with test data sets; however rule-based system was more useful when unknown or multiple faults were present.

A Programmable Logic Controller (PLC) is commonly used for process automation and equipment control in automated industries. Sekar et al. (2011) proposed an e-diagnostic approach for PLC based automated assembly systems. A dual robot assembly system was considered as an example and three levels of remote diagnostic architecture were employed. At Level 1, remote collaboration was established for remote connectivity to the robot. Level 2 had the same capabilities as Level 1 plus remote performance monitoring of operations. Level 3 provided advanced analysis and diagnosis with automated report generation. An extension of this work (Sekar et al., 2013) was the performance evaluation of remote trouble shooting system based on architectures, faults and skill level of operators. The study showed that there was no significant difference in terms of overall troubleshooting performance between an expert engineer and a novice operator.

Fault detection and classification using fuzzy logic was also examined by some researchers. An automatic fault diagnostic system based on fuzzy inference by analyzing the relationship between fault symptom and fault cause was proposed by Cheng et al. (2012). A pneumatic press was considered as the example. The fault symptom set and fault cause set were developed and related using fuzzy membership functions. The maximum membership principle was used to relate fault symptom vectors with the fault cause vectors. The approach proved to be more reliable, stable and faster than traditional fault diagnosis methods.

2.3 Machine Vision Based Methods for Inspection

Traditionally, human operators perform quality control and machine supervision tasks. Under certain circumstances, humans can do a job better than automated systems. But humans suffer from tiredness, stress and lack of concentration during repetitive tasks conducted over long periods of time. Human operators are also not suitable for quality check of a large number of small fast moving parts. Moreover, human operators

also require training to get the necessary level of expertise. MVI systems have the potential under these circumstances to obtain more precise inspection results in less time than humans.

Often thought to be the same, computer vision and machine vision are different terms. *Computer vision*, in the broad sense, is a field that includes methods for image acquisition, processing, analysis and understanding image contents across a wide range of theoretical and practical applications. *Machine vision* traditionally refers to the use of computer vision in an industrial or practical application or process to execute a certain function or outcome based on the image analysis done by the vision system (AIA Vision Online, 2016). It has been suggested that machine vision technology can help industry gain a competitive advantage in terms of a better product quality, high customer satisfaction, less inspection time and improved productivity (Malamas et al., 2003).

MVI systems use a digital camera and image processing software for an inspection of a product or process. The speed of inspection depends on the frame rate of the camera, speed of data transmission and time required by a PC to execute the algorithm. For simple machine vision applications a camera acquires images and transmits them to a PC that runs image processing software. The PC processes the images and outputs the inspection decision, with some delay, through a PLC. The delay is due to processing of the images outside the camera. Modern MVI systems use a smart camera for high speed vision inspection applications. The smart camera has its own memory and processor that runs an algorithm and processes the acquired images. A smart camera has a capability to handle the entire inspection application procedure including image acquisition and processing, making decisions based on the inspection results, and communicating these results to sorting mechanisms or to integrate with automation devices such as PLCs and robots. Smart cameras can be used as stand-alone devices or they can be connected to a larger network.

Web based remote monitoring, control and diagnosis of manufacturing processes using digital cameras, a PLC and PC based HMI was investigated by Yao (2005). The system monitored the factory via the internet and provided real time data regarding status of the machines. However, it was not an automatic fault diagnosis system as the collected information was interpreted by human operators. A second study (Hughes et al., 2012) proposed video feedback of machine faults to technicians for diagnostic purposes. They developed an intelligent system that captured and saved video data of fault occurrences on a laboratory conveyor that performed an assembly of three parts (a peg, metal puck and plastic puck). The system was controlled by a PLC and various sensors, and it was operated in a sequential fashion. If the events were not detected within a prescribed period of time, then fault alarms were generated by the PLC. Upon receiving the alarm, the fault monitoring system saved the most recent video frames into a file with a time stamp. Later, the video file could be accessed and reviewed by technicians and engineers for fault diagnosis purposes. In a third study a MVI for PCB defect detection, classification and localization using simple image processing techniques including template matching and morphological image segmentation was

investigated by Malge and Nadaf (2014). Their approach included simple image processing steps such as template matching, image addition, subtraction, and morphological operations. The algorithm successfully detected and classified PCB defects such as breakout, short connections, pin hole, wrong size hole, missing conductor and missing hole. This project proposes a PCB defect detection and classification system using a morphological image segmentation algorithm and fundamental image processing techniques.

ANNs are widely used for classification and pattern recognition problems using MVI systems. ANNs-based systems are considered "intelligent" because they have a learning component to them. They can also be applied to a variety of image processing tasks such as preprocessing, image compression, feature extraction, object classification and segmentation (Egmont-Petersen et al., 2002). The ANNs learn by examples. An application of ANN for object classification using the data extracted from images was implemented and compared with object classification using conventional image processing techniques by Huang et al. (1992). The results showed that although training time was long, ANN was more accurate and less sensitive to object orientation than conventional object classification techniques. MVI techniques are often combined with ANNs and fuzzy logic classifiers to achieve better classification accuracies. A machine vision system for the detection of missing fasteners on steel stampings was proposed by Killing et al. (2009). A neuro-fuzzy image classification algorithm was developed and compared with a threshold-based classifier. Images were captured, processed and feature attributes such as object size and color were extracted to train the classifier. It was reported that once optimized, the neuro-fuzzy classifier degraded in a less abrupt fashion when the input significantly deviated from the trained data. MVI techniques can be implemented in spatial domain or frequency domain. Spatial domain methods operate directly on an image pixels' gray values, while frequency domain methods modify the Fourier transform of an image using filtering process (Davies, 2005). Klein et al. (2014), investigated early bearing failures and diagnosis using image processing techniques applied to time-frequency representation (TFR) of vibration signals. A system with four stages was developed to detect and classify bearing failure signs based the vibration signals. In stage 1, TFRs of healthy machines were obtained using the methods such as wavelets and SIFT. These TFRs were used to create a baseline reference. Stage 2 calculated the distance between new TFRs and the baseline TFR. In stage 3, the distance TFR was analyzed using ridge tracking and other image processing techniques. Stage 4 established the relations between the detected ridges and the characteristic pattern of the bearing failure models for the classification.

MVI systems are often used for detection of unusual events in a scene such as event detection in crowded videos (Ke et al., 2007; Benezeth et al., 2009). These systems were developed for automated surveillance of an area, where they first model the normal behavior of the crowd using computer vision techniques like Spatiotemporal Volumes (STVs). The STVs are 3D representation of a scene in both space and time coordinates. During real time monitoring of the scene, new STVs are built at fixed interval of time

and compared with the normal STV. Any deviation form a normal behavior is identified and then isolated based on where in the model deviation occurred. These deviations were later used to classify the video data into a particular condition using the predefined rules. Szkilnyk et al. (2012) examined the application of a video event detection method, based on STVs, for fault detection in automated assembly machines. A trained model STV was modelled using a set of normal operation sequences. New STVs, for both normal and faulty operation sequences, were built and compared with the trained model STV, and classified into an appropriate category using a distance measurement. The number of video data sets used for normal operation sequence and faulty operation sequence were 65 and 20, respectively. The results were presented using a Receiver Operator Characteristics (ROC) plot and the inspection outcome was 89 % perfect classification with 6 % False Positive (FP, meaning the inspection system outputs a pass result but the real condition is fail) and 5 % False Negative (FN, meaning the inspection system outputs a fail result but the real condition is pass).

Automatic detection of stamping defects in lead frames (the core part of a semiconductor that functions as a base in ICs) using machine vision was implemented by Bhuvanesh and Ratnam (2007). The algorithm used image processing operations such as blob analysis, morphological operations and image subtraction to detect the stamping defects in the presence of transitional and rotational misalignments. The system was capable of detecting defects from both continuous reel and individual cut lead frames with rotational misalignment of +/- 10 degrees. Shahabi and Ratnam (2009) proposed a machine vision system for in-cycle monitoring of tool nose wear and surface roughness of turned parts. Image processing techniques such as filtering, segmentation and morphology were used to estimate the tool wear and surface roughness. The results showed that increasing the machining time of the tool decreased the roughness initially. However, after a certain time of machining (around 75 minutes), the surface roughness value increased due to the effect of growing notch wear. The maximum deviation of 10 % was recorded between the inspection using the MVI system and a stylus.

The following three methods will be highlighted as they are considered to be good candidates for further investigation for the work of this thesis.

Gaussian Mixture Models (GMMs)

Xiaokun and Porikli (2004) presented a novel approach to automatically detect highway traffic events, such as heavy congestion, high vehicle density with high speed, vacancy, traffic jam, etc., using a MVI system. The algorithm processed compressed videos of the inspection area, extracted event features from DCT domain without decoding the data, and classified traffic event using a Gaussian Mixture Hidden Markov Model (GMHMM). The GMM is a statistical technique of clustering data using probability density

estimations. Six traffic patterns were studied and the GMHMM was trained to model these patterns. The model detected events in real time with an accuracy of 94 %. Zezhi et al. (2014) used Gaussian mixture models for segmentation of moving road vehicles from the colored video data acquired using CCTV. The model then successfully classified vehicles according to their type (car, van and truck) and dominant color using a kernelized support vector machine.

This application had demonstrated that GMMs combined with blob analysis had the potential to solve O-ring machine fault detection problem. Transfer track jam on the machine has similarity with traffic jam on highways and missing hopper fault is similarly to vacancy on highways. The idea is interesting and it has the potential to solve O-ring machine inspection problem. Therefore, GMMs approach considered as the 1st potential candidate for O-ring machine inspection problem. However, the approach is not straight forward for the O-ring machine application due to a continuous rotation of small size parts (carriers and O-rings).

Optical Flow Method

Object detection and motion estimation from a video is a critical task in MVI systems for the analysis of moving objects. Motion estimation in image sequences using optical flow was first introduced by Horn and Schunck (1981) with the examples of a rotating sphere and a cylinder on its axis. Optical flow-based tracking methods are precise and reliable for the analysis of motion but there are several challenges associated with them such as computational burden, elimination of background movement and estimation of flow velocity. Patel and Shukla (2013) implemented a vehicle tracking algorithm using optical flow. The optical flow calculated using both Horn-Schunck and Lucas-Kanade methods was used to segment image frame for vehicle detection. The segmented vehicle was then tracked by plotting a rectangular bounding box around it in each frame. The velocity of the vehicle was determined by calculating the distance that the object moved in a sequence of frames with respect to the frame rate of video acquisition. The resulted were compared based on signal to noise ratio and angular error. Lucas-Kanade method had less angular error while Horn-Schunck method resulted in high signal to noise ratio. However, no publications were found that used optical flow for fault detection in machines.

This application had shown the potential that optical flow can be used for moving object detection and motion estimation. In case of the O-ring machine, carriers and O-rings followed specific path on the machine. The flow estimation of that path using optical flow technique would help in fault detection and classification. Hence, optical flow is the 2nd potential candidate to solve the O-ring machine inspection problem.

Running Average with Morphological Processing Method

Usamentiaga et al. (2013) proposed a new system to detect jams in a steel processing line at a crucial step of pickling using a MVI system. The proposed MVI system acquired images from the pickling line and processed them using the fundamental image processing techniques such as running average background detection, filtering, segmentation, blob analysis, and feature extraction. The extracted features were the measure of the density of the number of pieces ejected from the side trimmers. There was a direct correlation between the density of the features and a jam. These features were used to successfully detect jams in two nozzles using the running average method.

The MVI system was successful in detecting jam in the processing line during the 14 months of operation. This application had demonstrated that fundamental image processing techniques have the potential to solve the O-ring machine inspection problem. Therefore, running average approach is considered as the 3rd potential candidate to be considered for the O-ring machine inspection problem.

2.4 MVI Performance Measurement

MVI solutions are tailored to address specific application needs. The first criteria for development of any MVI system is to examine that whether it is possible at all to accomplish a MVI solution for a given task. If the answer is yes, then the second criteria is performance measurement. MVI systems use both hardware and software elements to solve a particular vision inspection problem. Hence, the overall performance of MVI systems depends on these two key factors. A MVI system consists of a digital camera, a lens and lighting. Camera features such as resolution, color space, sensor size, sensitivity and maximum frame rate at full resolution are selected based on the application requirements. Lens focal length is selected based on a distance between a camera and the part to be inspected. Lighting plays a vital role in a MVI system and its type, position and intensity determine the quality of the images. The selection of the correct camera and lighting is very important to achieve a desired goal with a MVI system because poorly acquired images are very challenging to process even with the help of state of the art algorithms.

There are many ways to solve a MVI inspection problem in terms of both hardware and software selection. When comparing different solutions, the important questions are:

- How can the performance of a MVI system be measured?
- Which solution is better among the possible solutions?
- How are different vision solutions compared?

This section of the chapter focuses on the above questions and answers them using specific application examples. Performance measurement criteria vary with MVI tasks and applications. Typical MVI tasks are *measurement, identification, verification, positioning,* and *defect detection* as shown in Figure 2-2. The author has created the figure by combining a material from multiple sources. The first column in the figure indicates MVI tasks, the second column lists applications and the third column states performance measurement criteria for MVI tasks.

Measurement tasks range from simple applications such as verification of an objects' presence and liquid level inspection in bottles, to complex applications such as measuring high-precision dimensional accuracies and geometrical tolerances on medical instruments. Typical applications of MVI for measurement are: checking dimensions of fasteners and bearings, verification of electrical connector dimensions, 3D profile measurement and weld quality inspection. The performance of a MVI system designed for measurement tasks can be specified in terms of precision, repeatability and accuracy. Typically, results are given in ± mm for linear measurements and in ± degrees for angular measurements. A high performance MVI should have high precision, repeatability and accuracy.

Identification tasks cover MVI solutions that involve reading 1D or 2D barcodes, labels, symbols and printed characters on products. Typical barcode reading applications are reading of barcodes on pharmaceutical products, vehicle disc brakes, printed circuit boards and packed food products. Printed

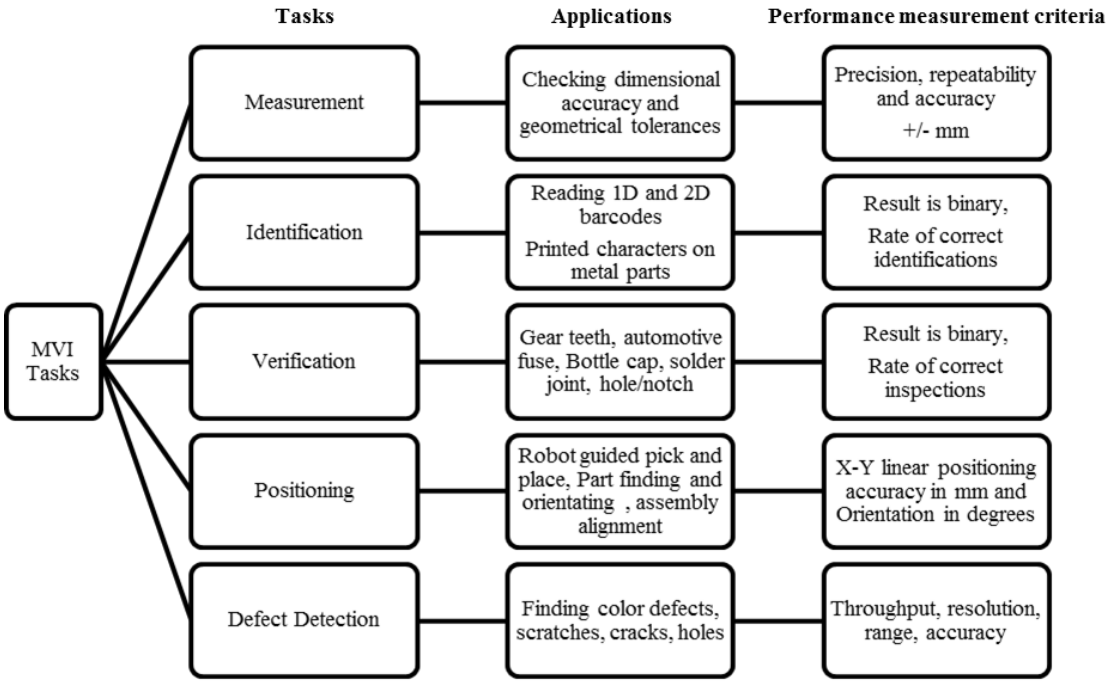


Figure 2-2 MVI tasks, applications and performance measurement criteria

character reading applications are reading of printed characters on labels/logos, cast metal parts and circular prints on bearings. The output of a MVI solution for identification is binary true or false. That means either the MVI system accurately identifies a printed code/label or it fails to identify it. The performance for identification tasks is measured in terms of the rate of correct identifications.

Verification tasks spans a broad application area for MVI solutions. It includes applications such as verification of parts, assemblies and packed goods. It may use the tools designed for other MVI tasks, such as measurement and identification tasks, to implement a complete product inspection system. A few applications of verification are: gear teeth verification, razor assembly verification, color verification of package food ingredients, bottle cap and safety seal verification, PCB assembly verification and feature (such as threads, holes and notches) verification on fasteners. The performance of verification tasks is reported in terms of the rate of correct classifications. The classification is considered as correct if it satisfies all inspection criteria specified in by the MVI algorithm.

MVI *positioning tasks* are related to determination of exact position and orientation of parts, finding and locating patterns, robot-guided picking and placement of parts. Some of the positioning applications are checking position and orientation of labels on bottles, verification of cell phone display location and alignment, finding patterns on solar cells, checking orientation of brake pads, guiding a robot for automatic picking and placement of PCB components, and alignment of automotive assemblies. The performance of positioning tasks for linear positioning is measured in mm and angular orientation measured in degrees. A high performance MVI positioning system should have high accuracy for both position and orientation.

Defect detection tasks use MVI systems for defect detection in products, machines and manufacturing processes. Tasks such as finding color detects, scratches, contamination, missing parts, incorrect assemblies, gaps and holes, fall into the category of defect detection. MVI algorithms look for pattern changes, texture changes and color variations for defect detection. Typical applications of defect detection using MVI systems are semiconductor inspection, weld inspection using X-rays, food and pharmaceutical products inspection for colors and texture, connectivity inspection on PCBs and quality inspection of rails. The performance of defect detection tasks is reported in terms of throughput, resolution, range and accuracy.

Even though MVI applications and solutions can be considered wide-ranging, performance measurement is usually based on at least one of the following approaches:

- Subjective approach
- Human expert approach
- Classification rate approach
- Confusion Matrix approach
- Receiver Operating Characteristic plot

- F- measure with Precision-Recall plot

These performance measurement criteria are explained further with examples given in the following paragraphs. For simplicity only one or two references for each performance measurement criteria will be given.

The easiest way to measure the performance of a MVI system is to use the subjective approach. For example, Heath et al. (1997) presented a way to measure the relative performance of edge-detection algorithms. Four edge-detection algorithms (Canny, Nalwa-Binford, Iverson-Zucker, Bergholm, and Rothwell) were tested on real gray scale images of complex scenes. Their performance was measured as a visual rating score by 9 participants. The Intra-class Correlation Coefficient (ICC) was used to measure consistency in the ratings by the participants. A high value of ICC indicated that there was very good consistency in the ratings. An average rating determined using a statistical analysis was considered as the overall performance measure. The four methods were compared based on that overall score and complexity in calculations. The proposed method of performance evaluation of MVS was purely subjective in nature and it has not been widely used.

Martínez et al. (2012) proposed a MVI system for defect characterization on transparent parts with non-plane surfaces that used the human expert approach. Three types of defects were considered in the paper: a) puncture defects such as bubbles, blisters and black points, b) surface defects such as excesses of varnish and c) linear defects such as threads and scratches. Image processing steps involved preprocessing followed by various segmentation and morphological operations. A series of tests using a commercial model headlamp lens were conducted. The performance of the system was measured in terms of the non-detection rate as obtained by comparing results with manual inspections by human experts.

As an example of the classification rate approach, Senthil Kumar et al. (2012) developed a new machine vision inspection system for the identification and classification of defects in MIG welding joints. ANNs with MVI approach was used to classify surface defects of butt joins into four categories: a) good weld, b) excessive reinforcement weld, c) insufficient weld and d) no weld. The inspection area was divided into four zones and features were extracted and input to a Bp type ANN. The network was trained with 80 patterns and classification results were presented in the form of a recognition rate obtained by the ratio of number of unseen patterns correctly classified to the total number of unseen patterns. The overall recognition rate of 95 % was achieved. Sun et al. (2014) presented an intelligent system for automatic inspection of thermal fuse using machine vision and ANNs. Four commonly seen defects, including black-dot, small-head, burr and flake were detected and classified using two supervised ANNs, namely, Bp network and Linear Vector Quantization (LVQ) network. The performance of both ANNs was compared

in terms of correct classification rate and processing time. The Bp network was slightly slower but provided higher classification rate (98.5 %) compared to LVQ (91.27 %) network.

As an example of the Confusion Matrix (CM) approach, Omid et al. (2013) designed and developed an expert egg grading system based on machine vision and artificial intelligence techniques. The features were extracted in the Hue- Saturation-Value (HSV) space and a fuzzy inference system based on triangular and trapezoidal membership functions was used for grading eggs based on their size and defects such as internal blood spots, cracks and breakages of eggshell. Information on 500 eggs was used to test the method. The results obtained from the fuzzy inference system were compared with results obtained from human experts. The performance was measured in terms of sensitivity and accuracy of the algorithm as obtained from the CM, which for a 2 x 2 matrix can be defined as:

$$CM = \begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix} \tag{2.1}$$

where TP, FP, TN and FN are the number of True Positives (TP) , False Positives (FP), True Negatives (TN) and False Negatives (FN), respectively. These numbers are obtained by comparing the output of the MVI system to the correct results as given by human experts:

- TP: the MVI system output is pass and the real condition is pass
- TN: the MVI system output is fail and the real condition is fail
- FP: the MVI system output is pass and the real condition is fail
- FN: the MVI system output is fail and the real condition is pass

The CM numbers can be used to calculate the measures of sensitivity and accuracy. Sensitivity is calculated using Equation (2.2). It is the ratio of true positive results obtained by the MVI system to the real pass results. It is the measure of how well the MVI system classifies a real pass result as pass. A perfect classifier should have 100 % sensitivity, which means zero false negatives.

$$Sensitivity = \frac{TP}{TP + FN} \times 100\ \% \tag{2.2}$$

Accuracy is obtained using Equation (2.3). It is the measure of how accurately the MVI system classifies a good condition as good and bad condition as bad. It is the ratio of the total true results to the total number of results. Accuracy of the perfect MVI system should be 100 %.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100\ \% \tag{2.3}$$

For any inspection algorithm with a binary output, the performance can be measured in terms of sensitivity and accuracy. When more than one algorithms are applied to the same application, the one with the highest accuracy and sensitivity is considered the best algorithm.

Szkilnyk et al. (2012) proposed the use of a video event detection method based on STVs for fault monitoring in assembly automation. STVs are 3D volumes that represent a video data of specific time length. Two levels of conveyor based automated assembly machine was used as the test bed. The system assembled plastic parts with metal parts. Five different faults such as dispenser mislead, jams and seized actuator were observed in the machine over a long continuous run. A single web cam recorded a set of image sequences for both normal and faulty operations. These sequences were then divided into two categories: training data and test data. STVs were calculated for a normal operation and STVs obtained with faulty data sets were compared with the normal STVs using a distance measure. The new STVs were classified as a normal or faulty STV depending on the deviation from the normal STV. The comparison between STVs was based on both partial matching and full matching. To evaluate the performance of the algorithm, the classification results were compared with the actual correct results. Using this information, the True Positive Rate (TPR) and False Positive Rates (FPR) were calculated using the following equations:

$$TPR = TP/(TP + FN) \tag{2.4}$$

$$FPR = FP/(FP + TN) \tag{2.5}$$

TPR is same as the sensitivity as mentioned in the previous approach. A plot of TPR versus FPR is known as a Receiver Operating Characteristic (ROC) plot, as shown in Figure 2-3. The ROC is a common way to illustrate the performance of a binary classifier. For any classifier its performance for a given data set is a point in the ROC space. The dashed line in the ROC plot is a random guess line. If the overall performance of the classifier falls on this line, it is a random classifier. That means that for any output of the classifier there is a 50% chance of its being true. Example of a random classifier is given as a point B in the figure. If the classifier's performance is below the random guess line, example shown as point C, then it is a poor classifier. Conversely, if the performance measure of the classifier is above the dashed line, it is a good classifier. The closer the point is to the upper left corner, the better the classifier. As an example, a classifier D is better than classifier A in the figure. The upper left corner point is the location of a perfect classifier.

In order to obtain the F-measure, one must first calculate the precision and recall numbers. Precision is calculated using Equation (2.6). It is the fraction of inspection results that are true positives. In other words, it is the ratio of TP decisions by the MVI to the sum of TP and FP decisions.

$$precision = \frac{TP}{TP + FP} \times 100\ \% \tag{2.6}$$

Recall is calculated using Equation (2.7). It is the measure of the fraction of TP. In other words, it is the ratio of TP decisions by the MVI to the sum of TP and FN decisions by it. Recall is same as the sensitivity and TPR as mentioned in the previous approaches.

$$recall = \frac{TP}{TP + FN} \times 100\ \% \tag{2.7}$$

The F-measure is the weighted harmonic mean (range 0 to 1) of precision and recall, and is calculated as:

$$F_\alpha = \frac{(1 + \alpha) \times precision \times recall}{\alpha \times precision + recall} \tag{2.8}$$

where α is a non-negative real value, representing the weighted coefficient between precision and recall. A higher value of F-measure indicates better performance of the classifier. As an example of the F-measure

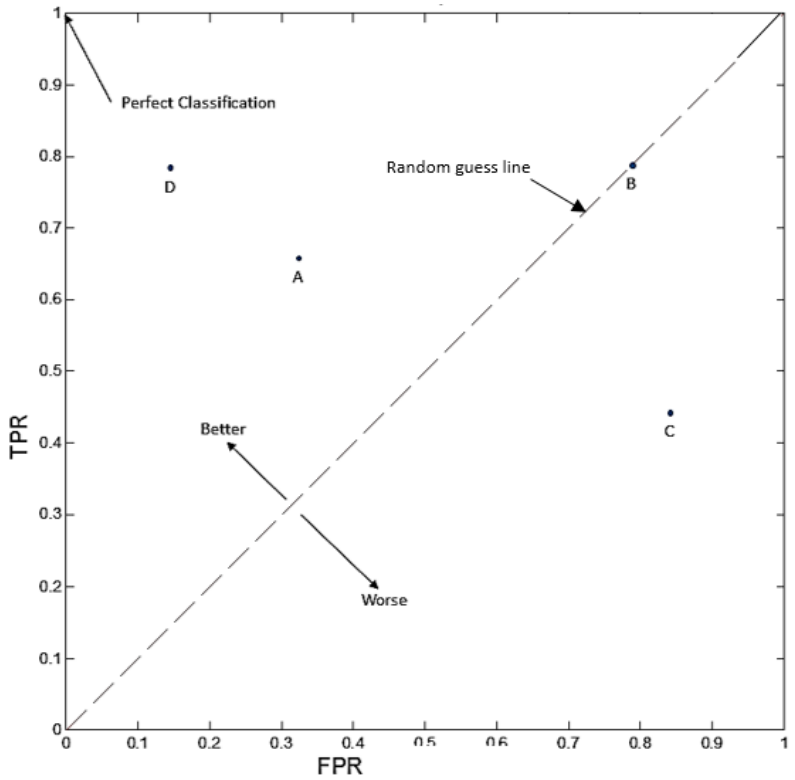


Figure 2-3 ROC plot for a binary classifier

approach, Shen et al. (2012) developed a MVI solution for bearing defect inspection that could inspect various types of defects on bearing covers, such as deformations, rusts and scratches. The approach was based on image processing steps for feature extraction such as average gray level, followed by defect detection with a Support Vector Machine. A final F-measure of 98% was obtained by the MVI solution.

The F-measure can also be reported in the graphical form as shown in Figure 2-4. It is the plot of precision versus recall. An F-measure closer to the upper right corner indicates better performance. In the figure, classifier A is a better classifier compared to classifier B because of its higher F-measure. ROC and precision-recall plots have some similarities and differences. They are similar in a sense that a line that results as a better classifier in ROC plot, also results as a better classifier in precision-recall plot. They are different in a sense that an algorithm that optimizes the area under the ROC plot line is not guaranteed to optimize the area under the precision-recall plot line (Davis and Goadrich, 2006).

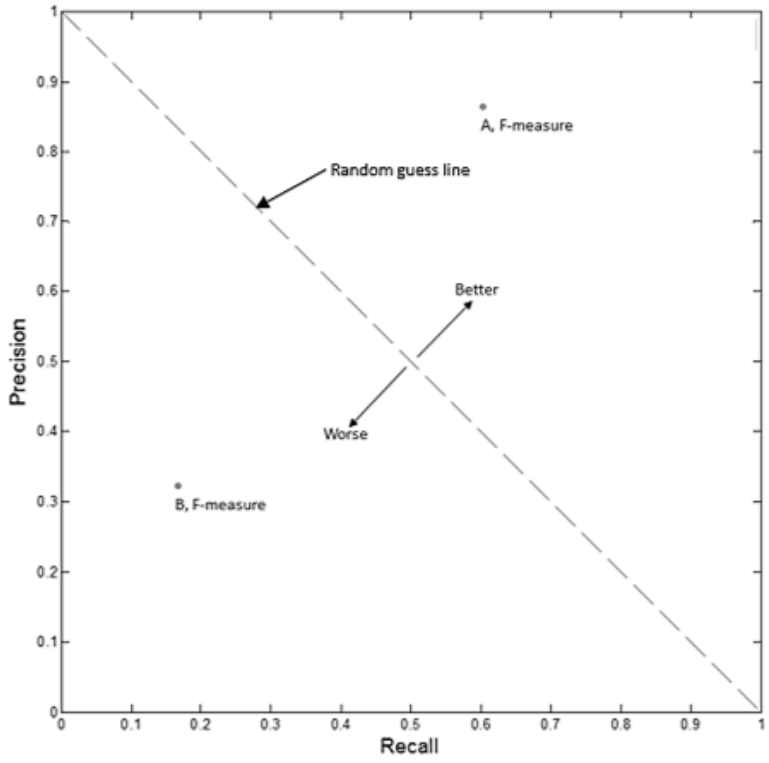


Figure 2-4 Precision - recall plot

2.5 Summary

Fault detection and classification of automated systems can be either non-machine vision-based or machine vision-based. Traditional solutions for non-machine vision-based methods include petri nets, behavioural models, parameter estimation, limit and trend checking, ANNs, GA and fuzzy logic. They are effective and efficient but they lack in terms of providing the information about the nature of the fault occurrence. These methods tend to detect the fault at the event of its occurrence, but they do not provide the insight of how the fault occurred. Machine vision-based methods have the potential to be better in that they can provide more information about the nature of fault using images of the actual events. MVI solutions can be based on computer vision techniques such as STVs, GMMs and optical flow or combined with other techniques such as ANNs and fuzzy logic.

The outcome of the literature review can be stated as follows:

- Examples were found of non-machine vision-based fault detection systems. Examples were found of computer vision applied to video and image analysis. But no examples were found of computer vision with video applied to fault detection in a machine. Thus, development of a MVI system with video for fault detection and classification appears to be novel.
- There are numerous ways to detect and classify faults using videos. The following three methods are considered to have the best potential (as explained in previous section) for the application at hand.
    1) Gaussian mixture models
    2) Optical flow method
    3) Running average with morphological processing method
- There are six accepted ways to measure performance: 1) subjective approach, 2) human expert approach, 3) classification rates, 4) sensitivity and accuracies with confusion matrices, 5) ROC plots and 6) F- measure with precision-recall plot. It is not clear which performance measure is best. This suggests the need for a more comprehensive performance index.

Chapter 3

Experimental Setup and MVI System Design

Automated assembly machines are designed to be operated round the clock to achieve a high production rate. Continuous operation of these machines results in wear of its mechanism that in turn leads to machine faults. Since, the machines are automated, minimum human supervision is used to monitor the machines. A MVI system can be used for automatic supervision of such machines for fault detection and classification. The MVI system monitors a machine with one or more industrial cameras. The cameras feed the video of the machine to a PC that runs image processing software. The PC detects any deviation in the machine's operating cycle from its normal operating cycle and warns an operator or stops the machine if a fault is being detected. Communication between the machine and a MVI system is carried out with a PLC.

An automated O-ring assembly machine is used as the test apparatus for this research project. The machine was originally designed and developed as a prototype by a Kingston based automated assembly machine builder. The machine separates and assembles a single O-ring from a bulk supply onto a plastic carrier. The machine was originally designed as a part of a larger machine in which the single O-ring was fed to the next stage in the assembly cycle for a coaxial cable. The machine uses a pneumatic system to transfer assemblies between stages. During a continuous run of the machine, several faults were commonly observed such as carrier jams, missing O-rings and inadequate air pressure. These faults have visual characteristics in the sense that they can be viewed by a camera. A MVI system can identify and locate visual faults from the Field of View (FOV) with various image and video processing techniques. This chapter describes the experimental setup and modifications implemented on the O-ring assembly machine as well as the design of the MVI system. Section 3.1 covers the description and operation of the machine. Commonly observed faults, the nature of the pneumatic system and introduction of controlled faults with a HMI-PLC is explained in Section 3.2. Detailed information regarding the design of the MVI system is described in Section 3.3. Finally, Section 3.4 summarizes the experimental setup and MVI system design.

3.1 O-ring Assembly Machine Operation

The O-ring assembly machine separates single O-rings from their bulk supply and assembles them onto continuously moving plastic carriers. An O-ring, its carrier and assembly are shown in Figure 3-1. The O-rings are of very small size (ϕ_p= 8 mm and T= 1 mm). A carrier has a groove on its surface into which an O-ring sits during the assembly process. The machine uses an electric motor to drive the machine. The motor is a SEW-EURODRIVE 0.5 HP motor rated at 1700 RPM (Rotations per Minute). The motor is connected to a gearbox with a ratio of 97.81:1. The rate of assembly is controlled by the RPM of the motor.

An Allen-Bradley PowerFlex 4 AC drive motor controller is used to run the machine at three different speeds. The motor controller uses two digital inputs to determine the motor speed. The machine does not run if both inputs are low and it runs at a speed if one or both of the inputs are high. The rate of assembly is proportional to the machine's RPM. Table 3-1 shows the three speeds of the machine with the corresponding rates of assemblies. The machine RPM has a range from 3.6 to 6.8, while the corresponding rate of assembly has a range of 58 to 110 assemblies per minute. The fluctuations in the rate of assembly for a fixed RPM is due to variations in the air pressure of the pneumatic system.

3.1.1 Machine Description and Modifications

Fault detection and classification with a MVI system for automated assembly machines is carried out on a modified assembly machine as shown in Figure 3-2. The main parts of the machine are two rotating transfer wheels (primary wheel and secondary wheel) that hold empty carriers and assemblies, a vibrating hopper as the supply of O-rings, a feed chute, two air transfer tracks (transfer 1 and transfer 2), an air-knife above the primary wheel for removal of unassembled O-rings and a vacuum system for collection of excess O-rings and assembled O-rings. The digital camera and 4 LED panel lights are also shown in the figure. As mentioned in the previous section, the machine was originally designed as a part of the larger machine,

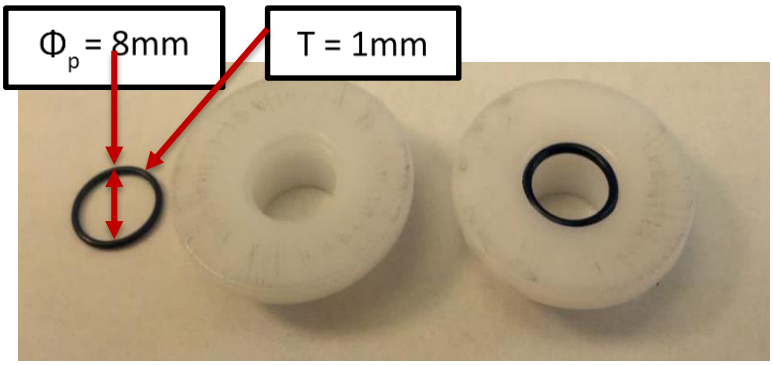


Figure 3-1 O-ring (left), empty carrier (middle) and assembly (right)

Table 3-1 Machine rpm and rate of assembly

Digital input 1	Digital input 2	Speed	RPM	Assemblies per minute
0	0	Stop	0	0
0	1	Low	3.60	58 to 60
1	0	Medium	5.10	84 to 86
1	1	High	6.80	108 to110

where single O-rings in their carriers were used for the next stage in the assembly procedure for coaxial cable. For the purposes of this research, the machine had to be modified so that O-rings could be separated from their carriers at the end of the assembly cycle and returned to the hopper. With this modification the machine could be run in a continuous fashion.

The primary wheel has 16 positions to hold carriers and assemblies. The maximum RPM of the machine is 6.8 RPM. Hence, the maximum rate of assembly can be calculated by:

$$rate\ of\ assembly\left(\frac{assemblies}{\min}\right) = 6.80\left(\frac{Rotations}{min}\right) \times 16\left(\frac{Assemblies}{Rotation}\right) \cong 108\ Assemblies\ per\ minute \quad (3.1)$$

This equation was the source of the numbers in Table 3-1.

3.1.2 Sequence of Operations for Normal Assembly Cycle

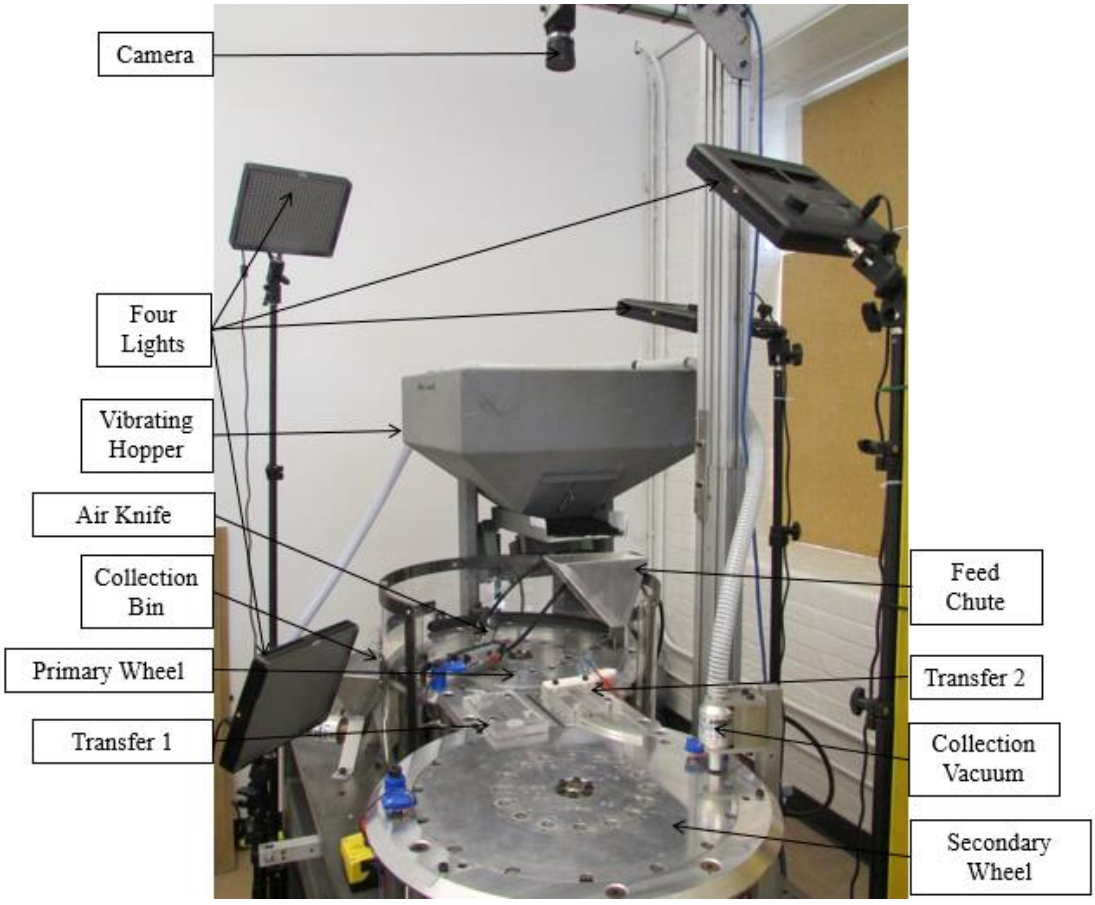


Figure 3-2 Automated O-ring assembly machine

The machine assembles black colored O-rings into continuously moving, white, circular carriers. The rate of assembly is 108 per minute at the maximum RPM of the machine. During operation, as the hopper vibrates, a steady stream of O-rings fall onto the primary wheel through the feed chute. The primary wheel has 16 slots to hold and transfer carriers/assemblies. As the O-rings fall from the feed chute, the aligning pins located beneath the primary wheel are raised and used to align the fallen O-ring onto the carrier. Here, a single O-ring is assembled onto the circular groove of a carrier. The primary wheel rotates at a rate of around 6.8 RPM that gives the linear velocity of 135 mm/s to the assemblies. The excess O-rings that are not picked up by a carrier fall onto the primary wheel and are blown off into the collection bin by an air-knife; where they are vacuum sucked and returned to the hopper. In the next stage of the assembly process, the carriers (each holding a single O-ring) are transferred, one at a time, to the secondary wheel via the air transfer track 1 (transfer 1). The secondary wheel, rotates the carriers to the collection vacuum valve. At this point, the O-rings are vacuumed off of the carriers and returned to the hopper. The empty carriers are then returned to the primary wheel via the air transfer track 2 (transfer 2) and the cycle is repeated. With this modification the machine could be run in a continuous fashion.

The machine uses a pneumatic system for vacuum suction and for the transfer of assemblies and empty carriers. The air for the pneumatic system is supplied by multiple compressors through a manifold. There are 6 electro-pneumatic valves; 2 for transfer tracks, 2 for air knives and 2 for vacuum valves along with 2 Line Vac valves for the collection vacuum. The basic sequence of operation is: 1) empty carriers appear on the primary wheel, 2) O-rings assembled onto the carriers with aligning pins, 3) assemblies transferred onto the secondary wheel and 4) O-rings recirculated back to the hopper. Any deviation from the normal sequence should be considered as a fault in the assembly cycle. During the normal cycle, assemblies move at different speeds at different locations on the machine. The assemblies move at the highest speed through the pneumatic transfer tracks, O-rings move at a relatively slow speed onto the primary wheel and O-rings fall from the hopper at a moderate speed. Under ideal circumstances, the machine should continuously assemble O-rings at the desired rate.

3.2 O-ring Assembly Machine Faults

During several runs of the machine for an extended period of time, a set of commonly occuring faults were observed. These faults occurred due to friction in the tracks, lack of O-rings flow from the hopper due to low vibrations and intermittent flow of O-rings, or improper operation of the air knives due to air pressure drop. Some of these faults could have serious consequences in terms of the machine's condition while other faults might have resulted in the lack of assemblies. The transfer track faults (transfer 1 jam and transfer 2 jam) were serious faults because they could damage the machine if immediate corrective action was not

taken. On the other hand, a lack of O-rings flow only results in missing O-rings on the carriers. This fault could have not damaged the machine but simply resulted in incomplete assemblies.

These faults are grouped into three Region of Interests (ROIs) with respect to the FOV of the camera. In the first stage of experimental video data collection, efforts were made to minimize the frequency of these faults. In other words, there was a need to run the machine under a normal operating cycle to get a measure of how well the machine ran under near fault free conditions. In the second stage of video data collection, a PLC with HMI was used to introduce controlled faults to acquire the video file for faulty operations.

3.2.1 Types and Locations of the Faults

The types and locations of the adopted set of four faults are shown in Figure 3-3. The faults were: transfer track 1 and 2 jams, the air knife fault and the hopper fault. Transfer track jams can occur due to low air pressure in the transfer track air nozzle, friction in the tracks itself or unassembled O-rings getting stuck in the narrow passage of the tracks. The air knife fault can occur due to inadequate air pressure that fails to

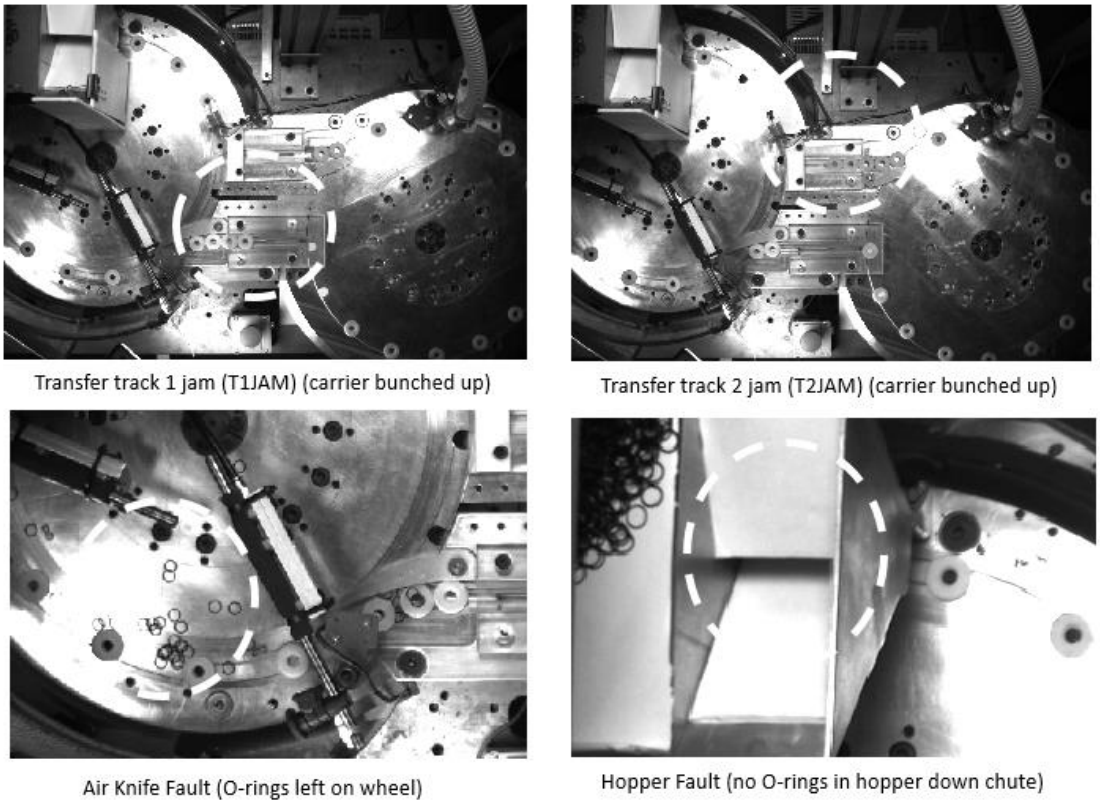


Figure 3-3 Types and locations of the faults

remove excess unassembled O-rings from the primary wheel. The lack of a continuous flow of O-rings from the hopper can occur due to friction in the hopper chute, insufficient amount of O-rings in the hopper or low hopper vibration amplitude. The hopper fault results in incomplete assemblies at the feed chute.

Figure 3-4 shows a schematic top-down view of the machine. The faults and their ROIs inside the FOV are shown in the boxes along with their labels. The flow of assemblies and empty carriers is also indicated by the arrows in the figure. During normal operation of the machine, gray arrows indicate regions where an O-ring should be present on the carrier, and white arrows indicate regions where the carriers travel empty. A stop button shown in the figure was used to stop the machine when necessary. The button resets the motor controller that stops the primary wheel from running and it switches the main air supply valve off. The machine can restart only after the motor controller is switched on manually. The faults are numbered from F1 to F4 and grouped into three ROIs. The adopted set of four faults with their description, symptoms and causes are tabulated in Table 1. Faults F1 to F3 might cause damage to the machine if the machine is not stopped soon after the fault occurs. Fault F4 will not damage the machine but it will result in the absence of O-rings on the carriers.

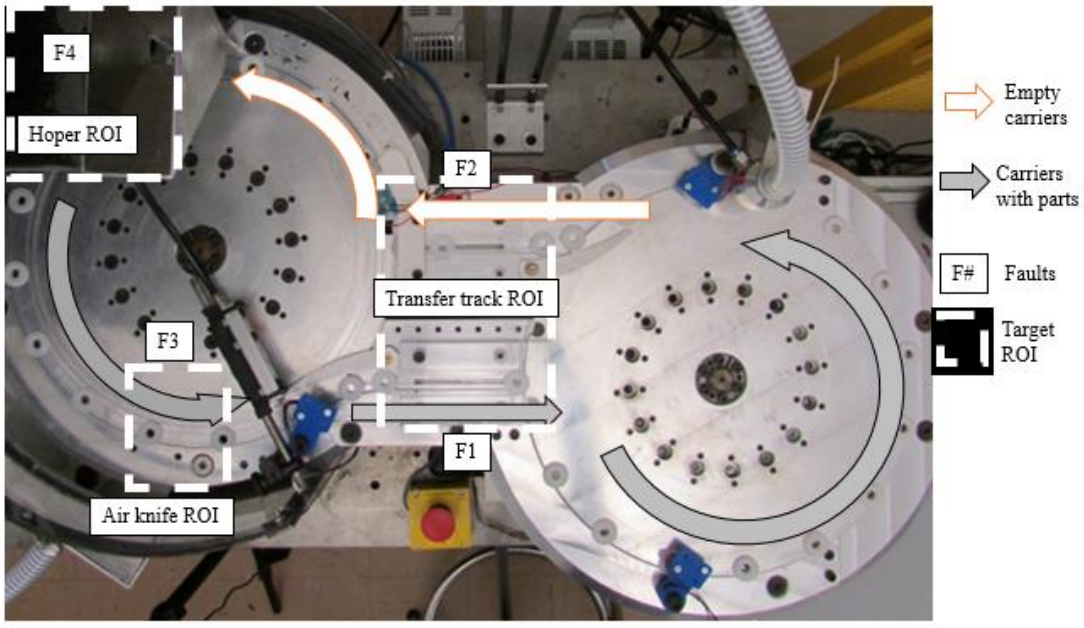


Figure 3-4 Locations 4 faults and 3 ROIs on the O-ring assembly machine

Table 3-2 Adopted set of faults for testing

Fault No.	Description	Symptom and Causes
F1	Transfer 1 jam	Loss of air to track, increased friction in air-track
F2	Transfer 2 jam	Loss of air to track, increased friction in air-track
F3	Air knife fault	Low air pressure results in presence of unassembled O-rings on the primary wheel
F4	Hopper fault	Empty hopper or low vibration results in the lack of O-rings flow from the hopper

The speed of movement of O-rings and carriers are different in the three ROIs seen in Figure 3-4. The carriers move at the maximum speed in the transfer track ROI, O-rings and assemblies move at a slow speed on the primary wheel in the air knife ROI and the O-rings fall at a moderate speed from the hopper in the hopper ROI. The speeds are calculated by measuring how much distance parts have travelled in a certain amount of time. The time is calculated based on the frame rate for video acquisition (30 frames per second). Figure 3-5 shows the distance travelled by O-rings and carriers in the three ROIs. These distances are used for the speed calculation, based on the maximum 6.8 RPM of the wheels, given in the next paragraph.

A carrier travels 120 mm in an average of 6 frames through the transfer track region. Hence, the speed of the carriers through the transfer track region is calculated from:

$$\frac{120}{1000} \times \frac{30}{6} = 0.60 \text{ m/s} \tag{3.2}$$

An O-ring rotates 13 degrees in an average of 14 frames on the primary wheel. The angular speed of rotation of the O-ring is calculated as:

$$13 \times \frac{30}{14} = 27.85 \text{ deg/s} = 0.48 \text{ rad/s} \tag{3.3}$$

O-rings rotate on a radius of 190 mm circle from the center of the primary wheel. Hence, the linear speed of the O-rings is given by:

$$\frac{190}{1000} \times 0.48 = 0.09 \text{ m/s} \tag{3.4}$$

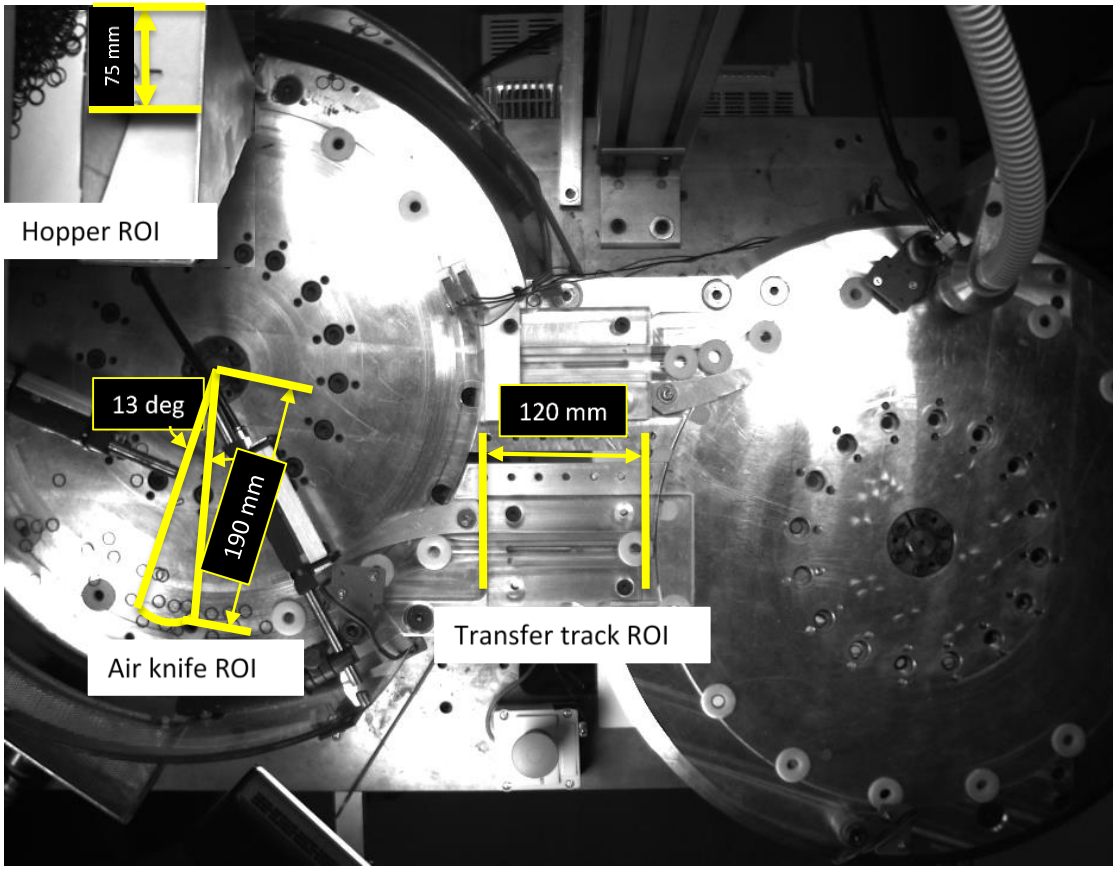


Figure 3-5 Distance travelled in the three ROIs

In the hopper ROI, the O-rings fall through the hopper and travel 75 mm in an average of 7 frames. Hence, the speed at which O-rings fall is obtained by:

$$\frac{75}{1000} \times \frac{30}{7} = 0.33 \text{ m/s} \tag{3.5}$$

Summary of the speed calculations is given in Table 3-3.

Table 3-3 Summary of speed calculations

Sr. no.	Fault area/ROI	Parts through the ROI	Speed in m/s
1	Transfer track	Carriers/assemblies	0.60
2	Air knife	O-rings	0.09
3	Hopper	O-rings	0.33

3.2.2 Pneumatic System for O-ring Assembly Machine

The pneumatic system circuit diagram for the O-ring assembly machine is shown in Figure 3-6. The O-ring assembly machine has 1 main electro-pneumatic air supply valve, 6 electro-pneumatic valves (2 for transfer tracks, 2 for air knives and 2 for vacuum valves) for various machine elements, 6 flow control valves, and 2 Line Vac valves for the collection vacuum. The pneumatic system receives air supply from three California Air Tools Portable Air Compressors (CAT-6310), each of which has a tank capacity of 24 liters. The compressors output air at a maximum pressure of 690 kPa through a Filter Regulator Lubricator unit. The output of the compressors are connected via manifold. A single line goes to the main air valve that delivers air to various parts of the machine.

The main valve is an electro-pneumatic valve that is controlled by actuating a solenoid with the PLC. The main valve is connected in such a way that in its default position it does not supply air to the machine but it supplies air only when the solenoid is activated. The output of the main valve is connect to 6 flow control valves that deliver air to the 6 electro-pneumatic valves. Air flowrate can be adjusted for different machine elements with the flow control valves. All 6 electro-pneumatic valves are 3/2 solenoid actuated spring return valves. Transfer 1 and transfer 2 solenoid valves are connected in such a way that in their default position, the valves supply air to nozzles in the transfer track. When a solenoid is activated, the valve spool changes its position and the air supply to the nozzle is cut off. Two Line Vac vacuum valves are used for recirculation of the O-rings. The vacuum valve on the primary wheel recirculates excess unassembled O-rings from the primary wheel into the hopper. The second vacuum valve is mounted on the secondary wheel and it is used to recirculate assembled O-rings from the carriers back to the hopper. They both receive air from two 3/2 solenoid valves that supply air in their default positions.

Two EXAIR air knives are mounted above the surface of the primary wheel for removal of dust, contamination and excess unassembled O-rings. An air knife is an effective and efficient system that projects a laminar (uniform) sheet of air along the entire length of the air knife to provide an air flow that covers the entire surface of the wheel. Both air knives are 10 cm long and are mounted at a 120 degree angle with respect to each other. They receive air from two solenoid valves. The right amount of air pressure at the air knives is very important because too low a pressure may not remove all unassembled O-rings from the primary wheel, while too high a pressure can cause the assembled O-rings and carriers to blow off the primary wheel. All electro-pneumatic valves require 24 VDC for activation of their solenoids. The main air valve of the pneumatic system is wired to the stop switch such that in case of a faulty operating condition, the main valve can be used to cut off air supply to the rest of the pneumatic system elements. All other valves are controlled by commands from the HMI-PLC.

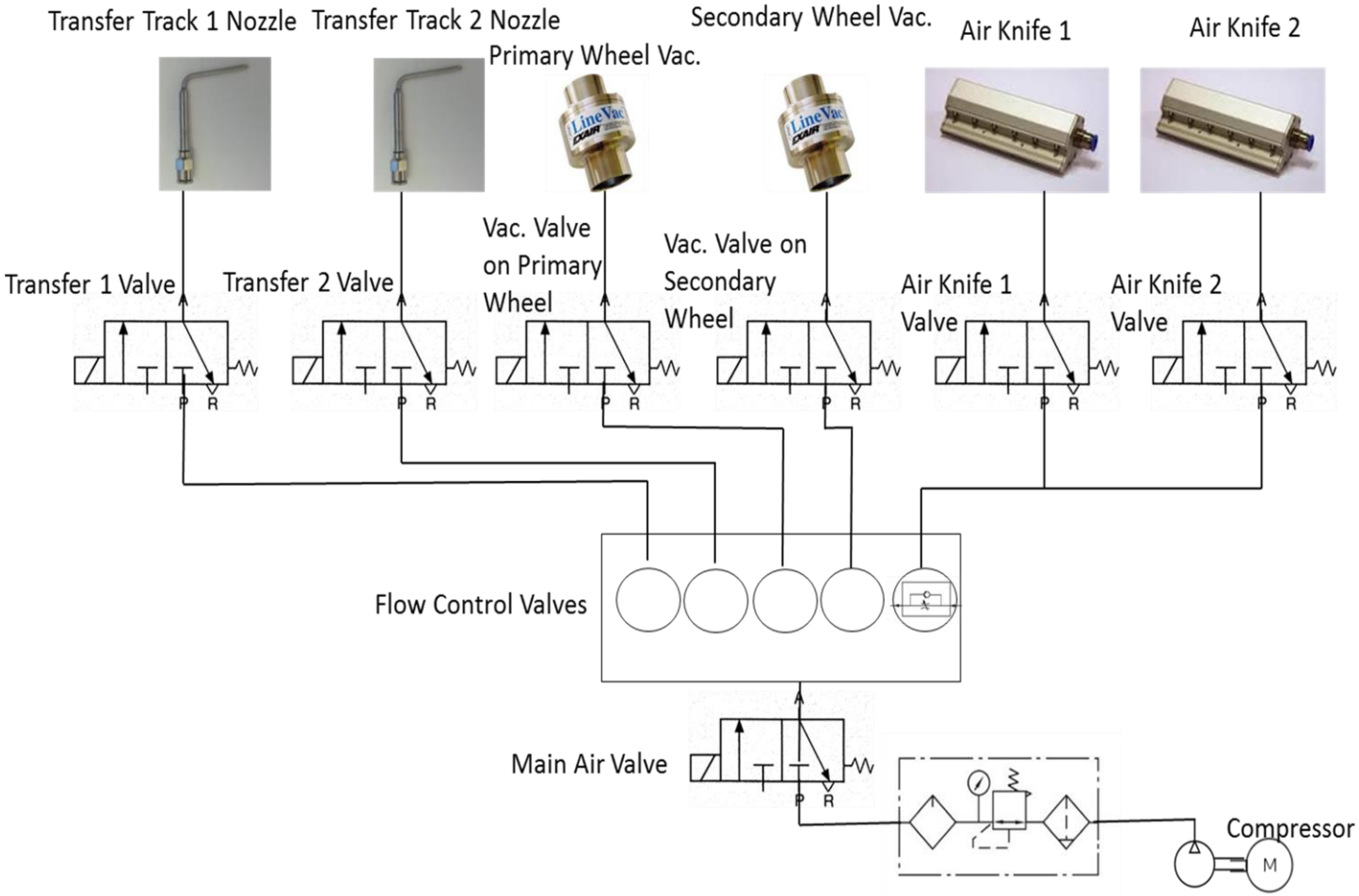


Figure 3-6 Pneumatic system for O-ring assembly machine

3.2.3 Controlled Faults with HMI-PLC

An Allen Bradley MicroLogix 1400 PLC (24 VDC, 20 inputs and 12 outputs) with a Panel View C600 HMI was used to introduce the controlled faults and to communicate the MVI system decisions to the machine (the detailed specifications are available in Appendix A). In the first step of each experiment, faults in the machine were minimized and the machine ran under a normal operating condition for 5 to 10 min. Faults were then introduced with the PLC and HMI as shown in Figure 3-7. The PC, HMI and PLC communicated with an Ethernet connection. Dedicated IP addresses were assigned to the PC, HMI and the PLC to avoid communication errors.

The PLC program is written as ladder diagram (given as Appendix B) with RSLogix 500 software from Rockwell Automation. The program reads the status of HMI buttons and the stop button, and outputs the signals to the various solenoid valves and the motor controller to control the operation of the machine. The faults are introduced with buttons on the HMI that communicates the button press decision to the PLC to turn on the appropriate solenoid valve. Figure 3-8 shows the PLC-pneumatic system connections and Figure 3-9 shows the fault selection screen on the HMI. The main air valve supply, motor controller and all other pneumatic valves are controlled by buttons on the HMI.

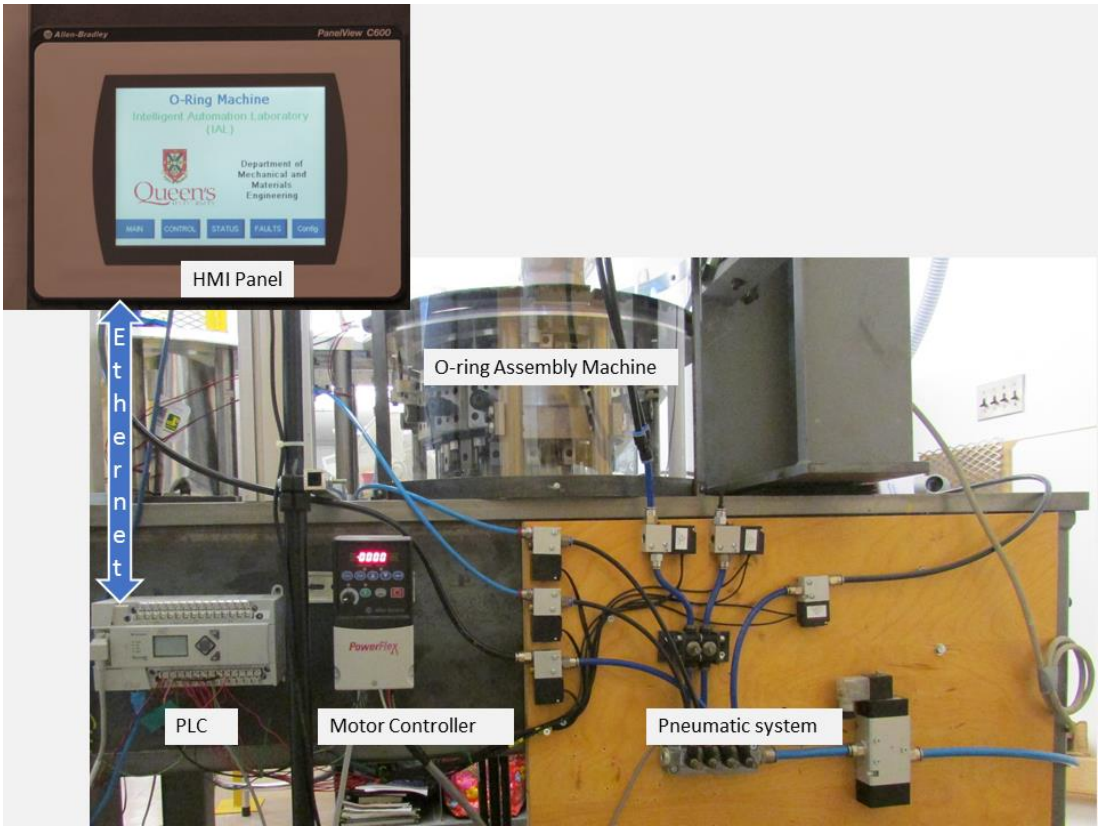


Figure 3-7 HMI-PLC and pneumatic system interface

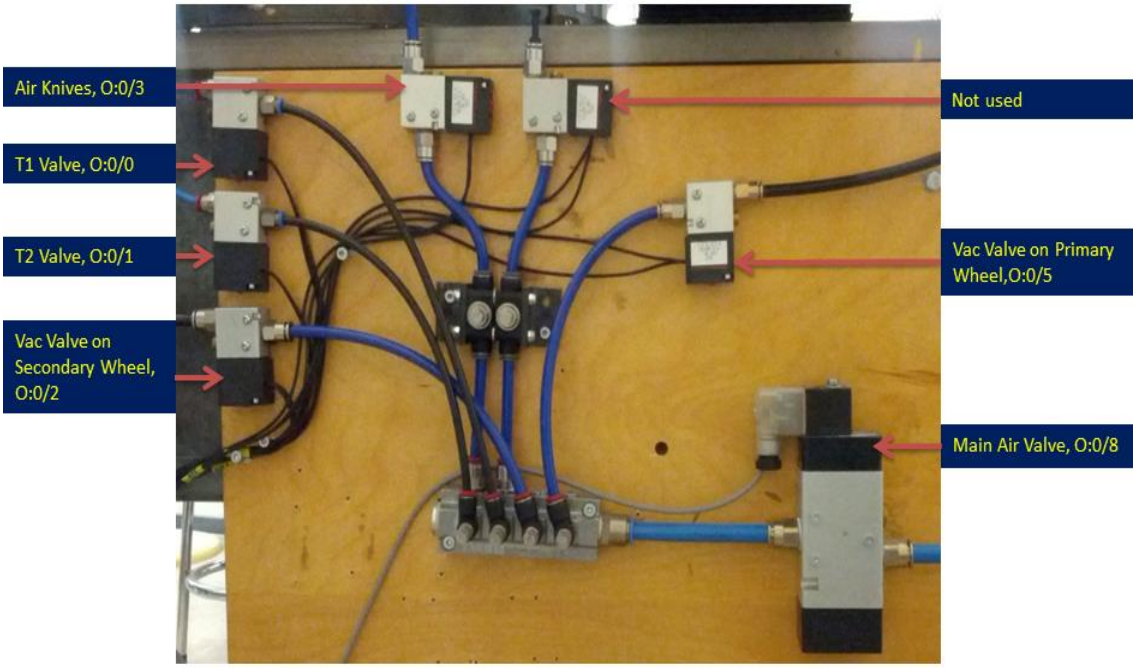


Figure 3-8 Electro-pneumatic valves and PLC outputs' connections

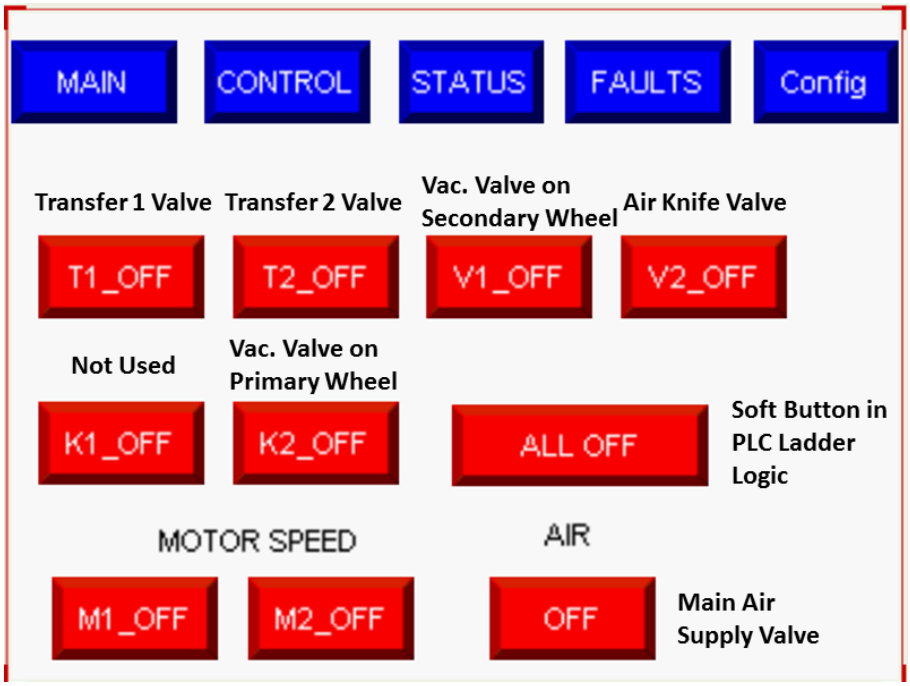


Figure 3-9 Fault control screen of the HMI

Figure 3-8 shows the connection between the PLC outputs and the solenoid valves. The corresponding PLC output module number is written beside each valve. For example, T1 valve, O:0/0 indicates that this valve's solenoid is connected to PLC's first output. All valves, excluding the main air valve, supply air to the system in their default state i.e. the state with a deactivated solenoid. Only the main air valve supplies the air upon activation of its solenoid. A fault can be introduced by activating a solenoid for the corresponding valve. For example, the air transfer track fault can be introduced by turning on the solenoid valve which supplies air to the transfer track. The valve switches its position when turned on and air is no longer supplied to the track which results in the transfer jam.

The HMI has five pages that can be accessed with the buttons on the top row of the screen as shown in Figure 3-9. The faults and the motor controller's operation are controlled with the fault control page of the HMI. There are 10 soft buttons on the HMI. To run the machine two button at the bottom right are to be pressed to activate the air supply to the system. The two motor speed button are used to set the motor at one of the three available speeds. Pressing only M1 or M2 runs the motor at slow and medium speed while pressing both M1 and M2 runs the motor at its maximum speed. All other buttons are kept depressed during a normal operation cycle. To record a video file with a faulty operating condition, one of the fault control button is pressed that activates the corresponding solenoid valve. The button stays in position once it is pressed. The configuration page of the HMI PLC provides information about the communication settings such as IP and node numbers. The fault introduction time was recorded and was used to determine how fast the MVI system detected the fault.

3.3 Machine Vision Inspection System Design

The elements of a MVI system are: camera with lens, lighting, image processing hardware, computer with image processing software and a PLC for communicating the decision. A typical MVI system example is illustrated in Figure 3-10. In this example, a conveyor belt carries parts to be inspected by the MVI system. A sensor is used to sense the correct position of the part with respect to the camera's FOV. Once the part arrives at the right location, the sensor triggers the camera to acquire an image of the part. The image is then transmitted to a PC through image processing hardware such as a frame grabber card. The image processing software runs an algorithm that inspects the image and outputs a pass or fail decision based on a predefined inspection criteria. If the image processing software's decision is pass, the part continues its journey on the conveyor and eventually falls into the accepted parts bins. If the decision of the inspection is fail then the PC signals an actuator through the PLC. The actuator extends and transfers the part to the rejected parts bin. The procedure is then repeated for the next part and the cycle continues.

The first step in the design of a MVI system is the selection of application specific camera-lens and lights. The quality of the original images is very important for any MVI system. There are two ways to get

high quality images: one way is a hardware fix, where the camera-lens and light settings are optimized to enhance the required features in the image; the second way is a software fix, where various image pre-processing techniques are used to improve the quality of the image. The efforts required for image enhancement after image acquisition depend greatly on the quality of the original images. Even with the help of state of the art image enhancement algorithms, it is hard to make better poorly acquired images. Selection of camera-lens and lights require several factors to be considered. The most important point to remember is that MVI systems are application specific, no single camera-lens and light combination can be used to solve all machine vision inspection problems.

3.3.1 Camera and Lens Selection

The camera and lens combination is selected first in the design of the MVI system. The camera selection factors are shown in Figure 3-11. The factors considered are nature of the digital interface, required resolution, sensor format, colors and type of sensor and a frame rate. Commonly preferred digital interfaces for industrial machine vision solutions are IEEE Firewire, Gigabit Ethernet and USB 2.0/3.0. IEEE Firewire has advantages such as low CPU load and low jitter but it has limitations of low bandwidth (32 MB/s for IEEE 1394a and 64 MB/s for IEEE 1394b). Gigabit Ethernet can support a long cable length (up to 100 meters) and has an easy infrastructure setup for multi-camera use. However, it has a high CPU load and price. USB 2.0/3.0 has become a standard in the consumer market and a lot of hardware supports it. Compared to USB 2.0, USB 3.0 offers nine times more bandwidth (up to 350 MB/s), better error management, higher power supply and longer cable lengths (up to 8 meters). USB cameras have less CPU

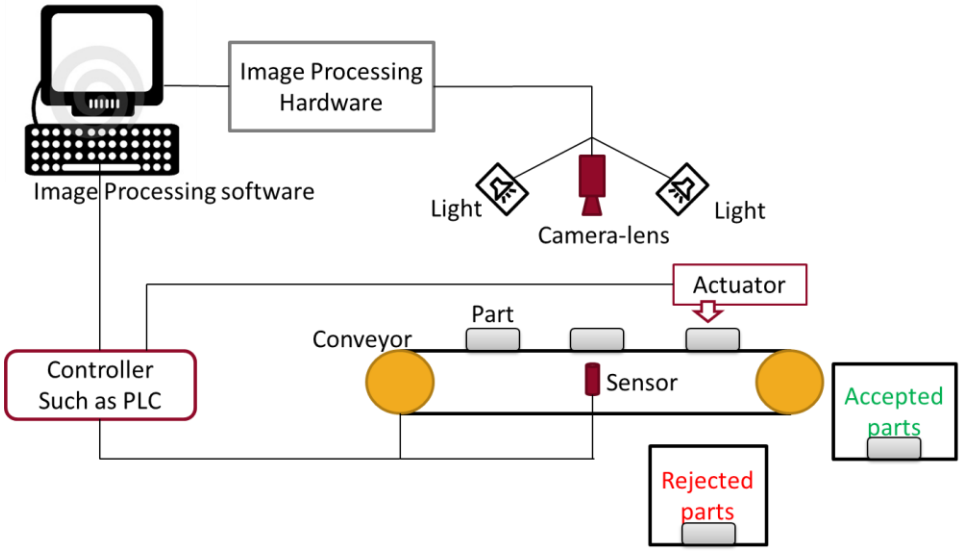


Figure 3-10 A typical MVI system

load compared to Gigabit Ethernet. Therefore, a USB 3.0 digital interface was preferred for the O-ring machine MVI camera.

Sensor resolution for MVI cameras can be broadly categorized into three segments: (1) less than 1.0 MP, (2) between 1.0 and 2.0 and (3) more than 2.0 MP. The higher resolution images look better but they need more processing time by the image processing software. A camera with more than 2.0 MP resolution would be appropriate for the project. Sensor format along with resolution decides the quality of image. A bigger sensor is better but it is more expensive as well. It was decided that a sensor with size 1/1.8" would fit the project requirement. A frame rate should be at least 10 times the rate of assembly. The O-ring machine assembles at a maximum rate of 108 assemblies/min or 1.8 assemblies/sec. Therefore, the frame should be at least 18 fps. The nearest available standard frame rate was 30 fps and it was selected for the project.

The next step after camera selection was setting the appropriate camera resolution, mounting the correct size lens and fixing the camera at a proper working distance. A single camera was used to detect multiple faults in the MVI system. The required FOV for the camera is the full image seen in Figure 3-4 and measures 710 mm by 381 mm. To achieve this FOV, a camera was mounted at the appropriate working distance, WD (Working Distance: distance from the front of the lens to the object under inspection), above the machine such that it can view this entire area. The following factors were considered while selecting a camera for the MVI system.

- Field of view

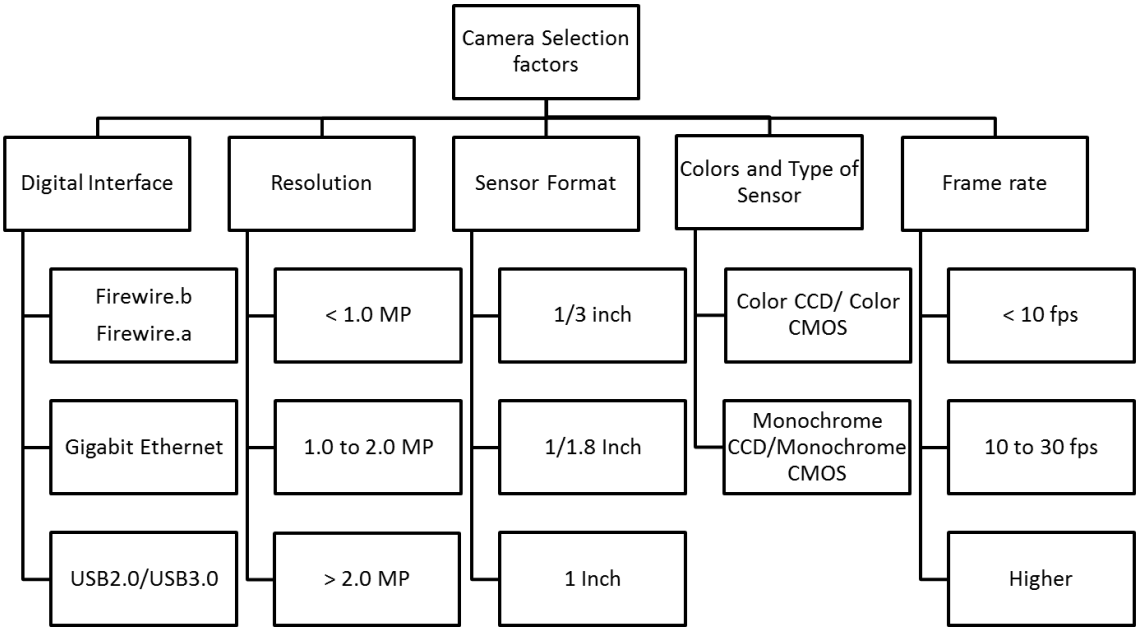


Figure 3-11 Camera selection factors

- Smallest feature
- Sensor resolution
- Frame rate
- Working Distance

If the FOV and the smallest feature to detect in the image are known, then the required camera resolution can be determined by the following equation: (National Instruments, 2014).

$$sensor\ resolution = 2\left(\frac{FOV}{smallest\ feature}\right) \tag{3.6}$$

The sensor resolution is same as the image resolution. The resolution of an image is the number of pixels (rows x columns) in the image. For accurate measurements, the smallest detectable feature should have at least 2 pixels. In the assembly machine, the smallest detectable feature was the width (1 mm) of the circular part. Using Equation (3.6) for the defined FOV and the smallest feature, the required sensor resolution was calculated as 1420 x 762 pixels.

Commonly used machine vision cameras come in standard resolutions, which vary from VGA (640 x 480 pixels) to UXGA (1600 x 1200), with the aspect ratio of 4:3. By comparing target sensor resolution to commercially available sensors, it was found that the target resolution fits in between XGA and UXGA. The higher resolution images take longer processing time while lower resolution reduces the FOV of the camera. The selection must be done by considering both FOV and required processing time (Wollak and King, 2009).

The WD is calculated using Equation (3.7). It depends on the Focal Length (FL) of the lens, maximum dimension of the FOV and corresponding camera sensor size (7.18 mm) as shown in Figure 3-12. The dimensions of different sensor sizes are given in Batchelor (2012). The WD decreases as focal length decreases and as sensor size increases. A shorter focal length gives a wider angle of view. The maximum FOV dimension was used for the calculation of the WD. Generally, lenses have fixed focal lengths and the WD is flexible in the application which gives many combinations of a lens and a camera for the same FOV.

$$WD = \frac{focal\ length\ \times FOV}{sensor\ size} \tag{3.7}$$

On the O-ring assembly machine, the WD was limited to a maximum of 1200 mm. A WD above this value caused interference in the FOV by the hopper. A lens with a short focal length caused distortions at the edges of the image. A suitable focal length was judged to be between 12 mm to 16 mm. A lens with

8.5 mm focal length and 2/3” format size was selected because it was a larger format lens for the smaller format camera which gave an equivalent focal length of 12 mm (as per Fujifilm USA, 2013). The use of a larger format lens for the smaller format camera helped to reduce distortions in the image at the corners because only the central portion of the view was captured by the camera sensor. Although, this lens and camera combination was not able to view the full FOV, it did cover the area of the machine that was needed to detect and classify the faults. In conclusion, the WD based on the available camera sensor sizes and lens focal lengths was calculated to be 1.1 m for a maximum FOV value of 710 mm.

Considering all the above factors and the project requirements, a Grasshopper3 2.8 MP Color USB3.0 Vision camera with 1/1.8” sensor and a resolution of 1928 x 1448 @ 26 fps from Point Grey Research Inc., as shown in Figure 3-13, was used as the MVI system camera.

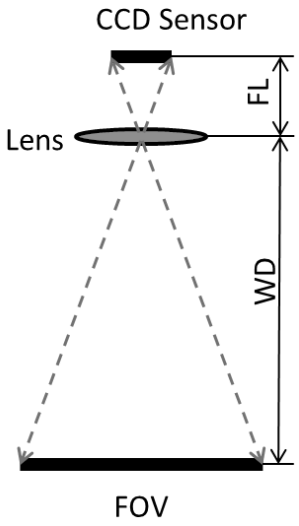


Figure 3-12 Camera sensor, lens and FOV positions

Figure 3-13 A USB 3.0 camera from Point Grey Research Inc. without lens

3.3.2 Light Selection

Light from the environment is undesirable for machine vision applications as it varies throughout a day. It is a common requirement for a MVI system to give accurate and precise results at any time irrespective of changes in the environment light (Miles et al., 2008). Therefore, MVI systems need controlled lights which can accentuate required features and minimize undesired features. The good performance of a vision application requires non-varying lighting conditions. Kopparapu (2006) presented an approach to obtain uniform illumination on the scene being imaged using several light sources. The use of correct lighting is crucial for the application related to surface parameter estimation with machine vision. Specular reflections from metallic surfaces can also significantly degrade the quality of the images.

Light selection is dependent upon the part size, part color, surface features, geometry, inspection environment, and the system needs. A full understanding of the immediate inspection area and the application requirements are the prime steps to be performed. Once a rigorous lighting analysis is done based on the above factors, a decision can be made to select a light solution from available options. Five factors, as shown in Figure 3-14, play an important role in the light selection based on a given application (Chauhan, Fernando and Surgenor, 2014):

a) Light source: From the wide variety of light sources commercially available for the purpose of illumination in machine vision inspection applications, three sources are very popular: LED, fluorescent lights and quartz halogen light with fiber optics. Due to flexibility, an extremely long lifespan, stability, low heat generation and cost effectiveness, LEDs are the first choice.

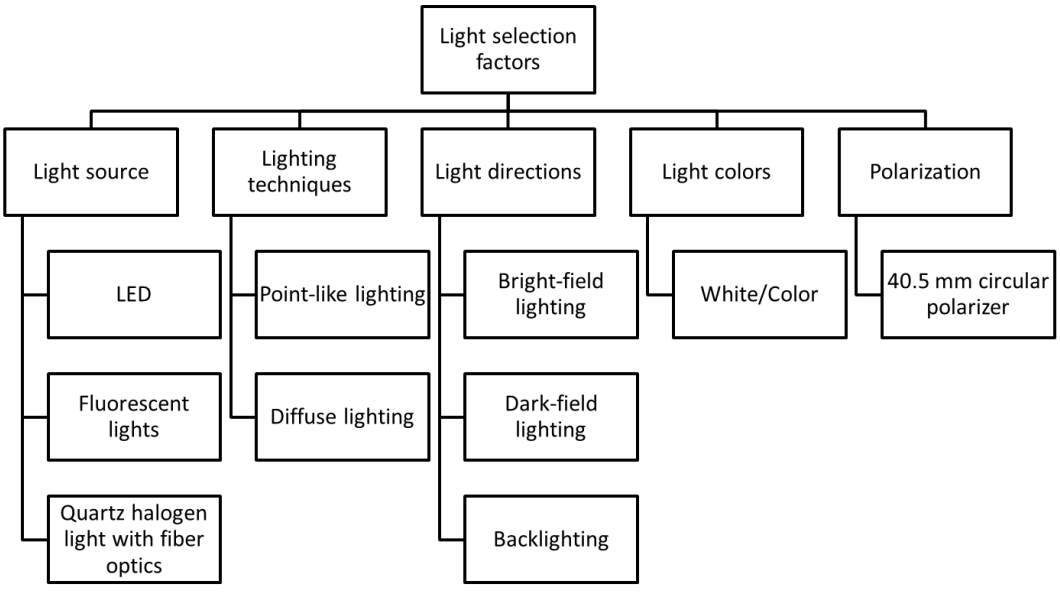


Figure 3-14 Light selection factors

b) Lighting techniques: Objects reflect light rays in two different ways; specular reflection and diffuse reflection. Polished and shiny surfaces reflect the light with specular reflections, which are bright but not stable with even the slightest change in the angle of a light. Diffuse reflections are from dull surfaces. Hence, they are dim but stable. In reality, objects may have a whole range of reflections between the specular and diffuse extremes (Braggins, 2000). For example, a PCB includes both metal pins (specular reflection) and plastic supports (diffuse reflection). Lighting techniques are based on a solid angle which is the angle of a unit sphere, centered on the object that the illumination occupies. There are two types of lighting techniques: point-like lighting and diffuse lighting. The lighting solid angle is small for point-like lighting and large for diffuse lighting. Incandescent lamps, optical fiber bundles, ring lights and LEDs are examples of point-like lighting. They are suitable for lighting dull parts and creating sharp edges. Specular surfaces are lit with diffuse lighting which covers a large solid angle around the object. Diffuse lighting minimizes glare and sensitivity to the surface angles on the parts. Fluorescent lamps are used for the diffuse lighting. The use of a diffuser in front of point-like light sources makes them diffuse. However, the diffuse lights make edges in the image blur and they are difficult to construct compared to the point-like light sources.

c) Lighting directions: The placement of a light with respect to an object and a camera decides lighting geometry. There are three basic categories of lighting geometry: bright-field, dark-field and backlighting (Microscan, 2014). As shown in Figure 3-15, the "W" shaped region describes the most basic lighting structure. The positions of the light field with respect to the "W" are clearly depicted. Bright-field is good for high contrast but it suffers from the specular reflections. One simple solution is to use a diffuser with a bright-field light. The dark-field light is placed outside the "W" region which eliminates specular reflections and highlights elevation changes. The flat background appears dark with the dark-field light. Backlight illumination is suitable for viewing translucent objects and silhouetting opaque objects. The light source is behind the object in backlighting.

d) Light colors: The interaction between a colored light and a colored object decides the contrast in the image. Like colors attract, hence same colored light will lighten a feature; conversely, opposite colors absorb which will darken a feature. Another advantage of using a monochromatic illumination is that the effect of any chromatic aberration in the lens will be reduced. Special filters can also be added to the lens for capturing the light within specific wavelengths.

e) Polarization: Metallic surfaces with high reflectivity affect the accuracy in a typical feature recognition and measurement application because the shiny objects result in the glare in the image (Zorcolo et al., 2011). A polarizer can be used in such cases to eliminate specular reflections. The polarizer is placed in front of the camera lens. Light that is specularly reflected from a shiny object is blocked by the polarizer

resulting in reduced spot of glare. The noticeable drawback to this method is that there is less light transmitted from the lens, which results in a dark image. A TIFFEN circular polarizer with size 40.5 mm was used to minimize the bright reflections from the shiny wheel surfaces on the O-ring machine.

By considering the above factors four Aputure Amaran AL-528 LED panel lights with adjustable intensity were selected as the lighting solution for the MVI system, as shown in Figure 3-16.

3.3.3 Light and Camera Positioning

The automated assembly machine has both dull (plastic) parts and shiny reflective (aluminum wheel) surfaces. The specific challenges are the small size of the O-rings, the influence of the environmental light, holes in the carrier and the shiny nature of the surface of the metallic primary and secondary wheels. Using the guidelines in the previous section, LED panel lights were selected for illumination. The position of the lights and camera relative to the FOV was the next important factor to be considered. Hence, an algorithm was tested with different lighting positions. Once a camera view and the lens focal length were set, tests were conducted under six different lighting conditions; 1) ambient light, 2) diffuse light, 3) bright-field light with a diffuser, 4) bright-field light with a polarizer, 5) dark-field light with a diffuser and 6) dark-field light with a polarizer. Carriers and O-rings detection from images acquired under an ambient light and dark-field light with a polarizer are shown in Figure 3-17.

The O-rings assembled onto the carriers were black and the carriers were white. The carriers had hole at the center due to which the carrier with and without O-ring look similar (white circle with a small concentric black circle) in the image. Using only the overhead lights, it was not possible to detect the O-rings inside the carriers. Therefore, the area beneath the wheels was covered with a white foam board that reflected the light and provided the effect of a backlight. Once the system was set up, a few preliminary tests were conducted to determine how well the system detected the carriers and O-rings. The tests were

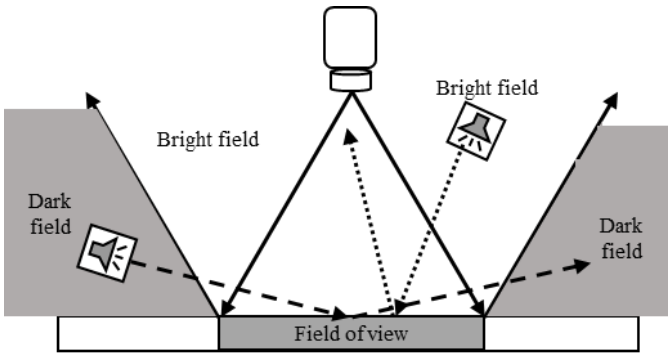


Figure 3-15 Positions of bright field and dark field lights

Figure 3-16 Aputure Amaran LED light with available diffusers

conducted under the six lighting conditions and results were used to make a decision regarding the proper positioning of the lights.

During initial trials, images were acquired under ambient lighting condition and processed to detect the carriers and O-rings. A background image was first captured without the presence of any carriers and O-rings. Each successive image frame with the carriers and O-rings was then captured and processed with a histogram equalization technique to minimize the effect of variations in the lighting. Next, a background subtraction was carried out for all the images in which the background image was subtracted from the image with the carriers and O-rings. Circle detection was then applied to the difference image to detect the carriers ("bright circles" in the image), and the negative of the difference image was processed to detect the O-rings ("dark circles" in the image) as shown Figure 3-17. The circle detection required predefined range of radii for both the carriers and O-rings that was calculated from the images using a distance measurement function.

Variations in the ambient light and the small size of the O-rings made circle detection difficult. Further, it was hard to differentiate between carriers with and without the O-rings. To mitigate these problems, LED panel lights were set on four sides of the machine and a foam board was fixed under the wheels to get the effect of backlight for O-rings detection though the holes in the carriers. This resulted in the O-rings clearly appearing as black circles on a white background. The camera was mounted 1.1 m above the wheel surfaces and a white diffuser was fixed on the light to reduce specular reflections. The performance of the algorithm was measured in terms of a detection ratio. It is the ratio of the number of objects detected from the image to the actual number of objects present in the image. The ratio was calculated for both carriers and O-rings separately.

The external lighting helped to "see" the carriers but there were false detections in the output due to the presence of other circular objects, such as bolts and holes. The parameters in the algorithm were set

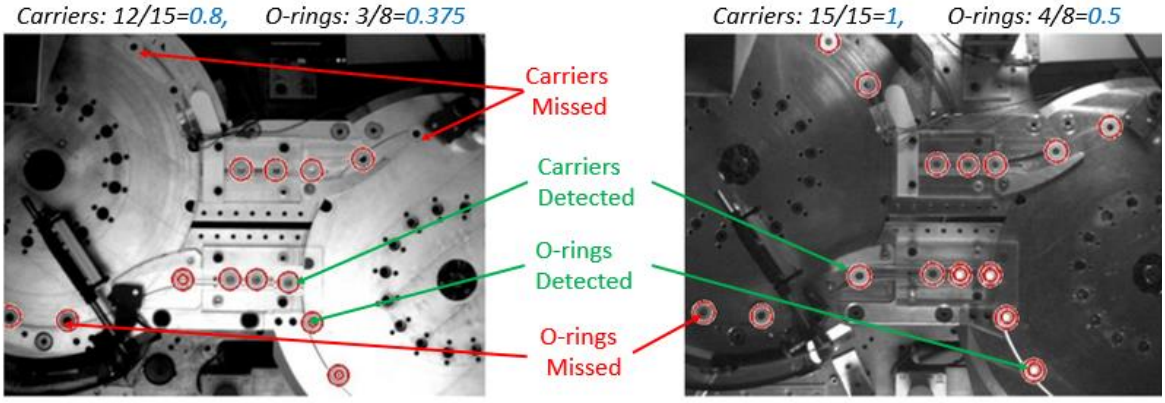


Figure 3-17 Image (left) under ambient light and image (right) under dark-field with a polarizer

to prevent false circle detections. The results of the tests are presented in the form of a bar chart shown in Figure 3-18.

The dark-field light with a diffuser resulted in the glare free images but the intensity of light was not sufficient to detect the O-rings. The high intensity of the dark-field light detected more O-rings but led to the glare in some regions in the FOV. To compensate for this high intensity of the dark-field light, a polarizer was used with a lens. Therefore, the best results (100 % detection ratio for the carriers and 50 % detection ratio for the O-rings) were obtained with the dark-field light and the polarizer. The O-rings were still difficult to detect at all locations because they rested in a circular groove on the carriers; hence it was difficult to illuminate them using the LED panel lights. The O-rings were successfully detected from the region of a backlight from the foam board, placed immediately after the transfer track underneath the secondary wheel. The reflected backlight covered only limited portion of the FOV, hence the detection ratio for the parts was lower than that of the carriers.

As a final setup, four LED panel lights were placed around the machine to obtain a dark-field lighting effect for the wheel surfaces and with the foam board under the wheels to obtain a backlight effect for the holes in the carriers. The camera was mounted at 1.1 m above the wheel surfaces and had the polarizer above the lens. Figure 3-2 illustrates the adopted positions for the four LED panel lights.

3.4 Summary

The experimental setup for fault detection and classification with a MVI system was explained in this chapter. An industrial O-ring assembly machine was considered as a test bed for the experiments. The machine assembles up to 108 assemblies per minute. The machine was modified to recirculate assembled

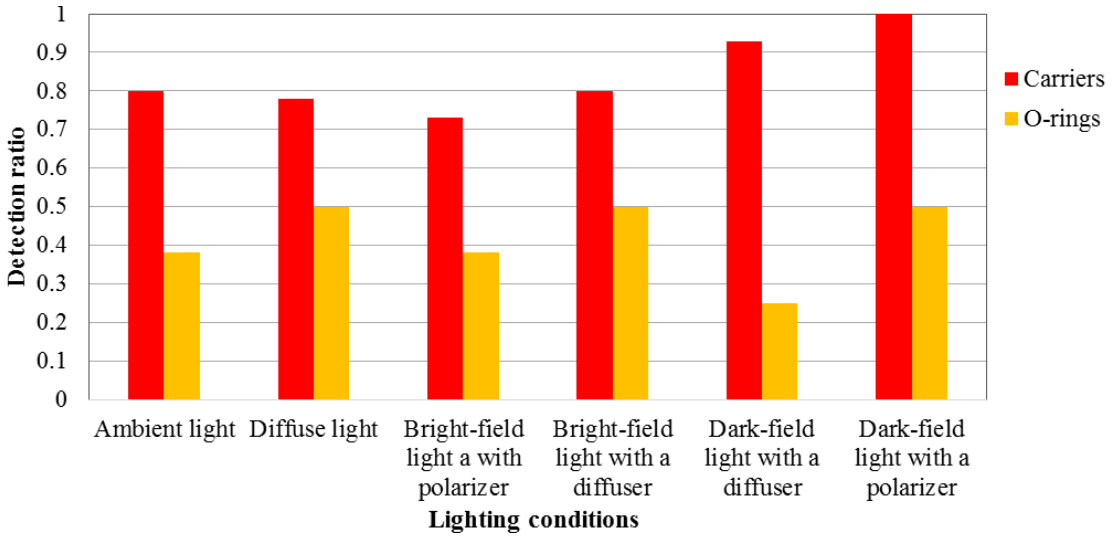


Figure 3-18 Detection ratio vs. lighting conditions

O-rings for continuous operation. The normal assembly cycle was explained. The adopted set of four faults, their types and locations, were discussed. The Introduction of controlled faults using a PLC with an HMI was explained. The importance of the camera and lens selection, lights selection and their relative position was discussed.

It was determined that a camera with an UXGA resolution and a minimum capture rate of 18 fps was required to observe the entire target FOV. The high reflectivity of the machine surface and a small size of parts made the inspection difficult under ambient light. Therefore, external controlled LED panel lights were used and the effect of change in the light conditions on the detection ratio was studied. It was observed that given the high surface reflectivity, a dark-field light with a polarizer provided the best results. The detection of the carriers was relatively easy compared to the O-rings due to their size and bright white color. The most challenging task was to detect the O-rings assembled onto the carriers. The LED panel lights did not directly help in the detection of O-rings. A backlighting effect obtained by using white foamcore board as background material, which helped in the task of O-ring detection. The specifications of the machine, HMI-PLC and MVI system are given in Table 3-4. The detailed specifications are available in Appendix A. The PLC program for machine control and fault introduction is provided in Appendix B.

Table 3-4 O-ring Assembly Machine experimental setup specifications

Component	Description
Drive and Control System	
Motor	▪ SEW-EURODRIVE, V:208/360 , 60 Hz, 2.21/1.28 A, 0.5 HP, 1700 RPM ▪ Gear box ratio: 97.81 : 1
Motor Controller	▪ Allen –Bradley PowerFlex 4 AC Drive (Cat. No: 22A-V2P3N104) ▪ Input : 100 – 120 V AC (±10%) – 1-Phase || Output: 0 – 230 V 3-Phase 2.3 A
PLC	▪ Allen-Bradley® MicroLogix™ 1400 (Model 1766-L32BXB) ▪ 24 VDC (20 Inputs, 12 outputs)
HMI	▪ Allen-Bradley PanelView C600
Pneumatic System	▪ California Air Tools Portable Air Compressor CAT- 6310, Ultra Quiet & Oil Free, 110 V, 1 HP, 6.3 Gal (quantity: 3) ▪ 3/2 electro-pneumatic valves (quantity: 7)
MVI System	
Camera	▪ Grasshopper3 2.8 MP Color USB3 Vision (Sony ICX687), 1/1.8” ▪ 1928 x 1448 @ 26 fps, Being used at 964 x 724 @ 30 fps
Lens	▪ Pentax C30811TH-(C815B) ▪ 8.5 mm Focal Length, 2/3” format
Lights	▪ Aputure Amaran AL-528W LED Light (quantity: 4) ▪ 30W, 2.8 A, 5500 K , Life 100000 hours
Software	▪ MATLAB R2014a

Chapter 4

Video Data Acquisition and Methodology

Fault detection with a MVI system for automated assembly machines is carried out on a modified O-ring assembly machine. Three MVI based methods are developed for detection and classification of the machine's operating conditions. The machine uses a pneumatic system for transfer of assemblies, removal of excess O-rings from the primary wheel and recirculation of the unassembled and assembled O-rings. A set of four individual faults has been selected for testing purposes. The MVI system should detect and classify these faults from their video data only. The locations of the faults along with their corresponding three ROIs are shown in Figure 4-1. This figure is repeated from Chapter 3 to support the methodology description. The faults F1 to F4 are the set of faults listed in Table 3-2.

Video data sets are acquired for each operating condition using an overhead camera mounted 1.1 m above the surface of the wheels. Three MVI methods: (1) fault detection with GMMs and blob analysis, (2) fault detection with optical flow estimation and (3) fault detection with foreground running average area are tested with each video data set for their ability to detect and classify the adopted set of faults.

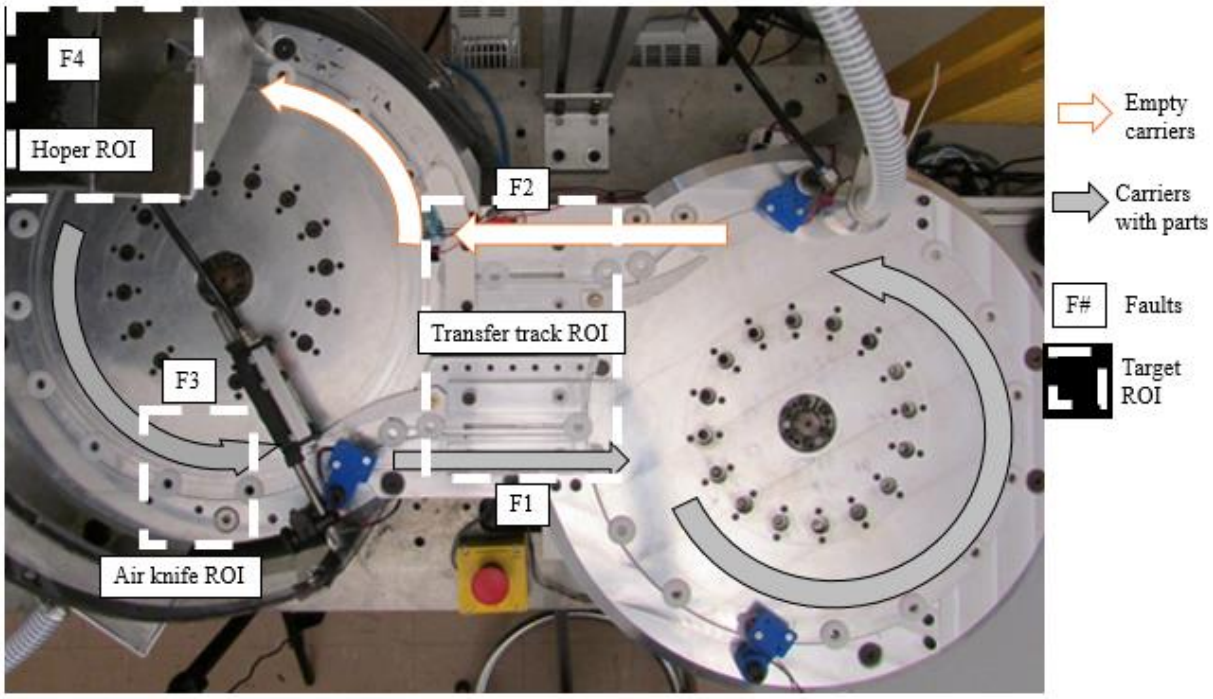


Figure 4-1 Fault locations and ROIs on the O-ring assembly machine (also given as Figure 3-4)

This chapter presents the data acquisition and methodology used to carry out the experiments. Section 4.1 outlines the objectives of the experiment. Section 4.2 explains the procedure for video data acquisition. The video data is acquired for both normal and faulty operating conditions of the machine. The controlled faults are introduced with the HMI-PLC. Preliminary data acquisition was performed with an IEEE 1394 digital camera and LabVIEW image processing software. It became immediately apparent that an expansion of the ROIs was required to see all faults, which in turn meant one needed a faster and better quality camera. Therefore, the high speed USB 3.0 digital camera identified in Chapter 3 with MATLAB software was used for all experiments reported in this thesis. Section 4.3 describes the concept, theory and algorithm development for fault detection with GMMs and blob analysis. Section 4.4 covers the theory and algorithm development for fault detection with optical flow estimation. Fault detection with foreground running average area and the algorithm development is covered in Section 4.5. Section 4.6 summaries the data acquisition and methodology used for the experiments.

4.1 Experiment Objectives and Strategy

The main objective of this thesis was to design and develop a MVI system for automatic detection and classification of faults in automated assembly machines. To prove the concept, an industrial O-ring assembly machine was considered as a test bed. The machine had been modified for the purposes of this research. Chapter 3 described the experimental setup and modifications done to the O-ring assembly machine. The other two objectives were to test three intelligent MVI methods for fault detection and classification from video data sets and compare their performance. Three MVI methods for fault detection are explained in this chapter. Each method is tested with four types of faults, namely transfer track 1 and 2 jam, air knife fault and hopper fault. To achieve these objectives, the following procedure was adopted for the experiments.

1. Acquire normal operation video data from the O-ring assembly machine with an overhead camera under controlled lighting conditions.
2. Introduce pre-observed faults by controlling corresponding electro-pneumatic valves with the HMI-PLC and acquire video data of faulty operation.
3. Develop three MVI methods for fault detection and classification on the O-ring assembly machine.
4. Test each MVI method with all acquired video files and record results of the inspection.
5. Evaluate each methods' performance with the pre-defined performance criteria and highlight strengths and limitations of the methods.

A GUI (Graphical User Interface) was developed in MATLAB (MATLAB, 2014b) software for each method with the indicators of the machine's operating condition. The GUI with a sample frame and three ROIs is shown in Figure 4-2. The GUI processes the video data sets and detects the machine's operating condition. The MVI methods do not require separate training and testing data sets but they learn the normal behavior of the machine from the first few frames of each video. Later, it classifies the machine's operating condition by comparing the future video frames with the normal video frames. Features are extracted from the video frames that represent the machine's normal behavior. If for a set of frames, the features exceed their normal or reference value, the condition is considered as a fault. Depending on the machine's operating condition, one of the indicators will be highlighted. All methods were tested with each type of fault. But as an example, only the transfer track operation is considered and explained with reference to all MVI methods in this chapter.

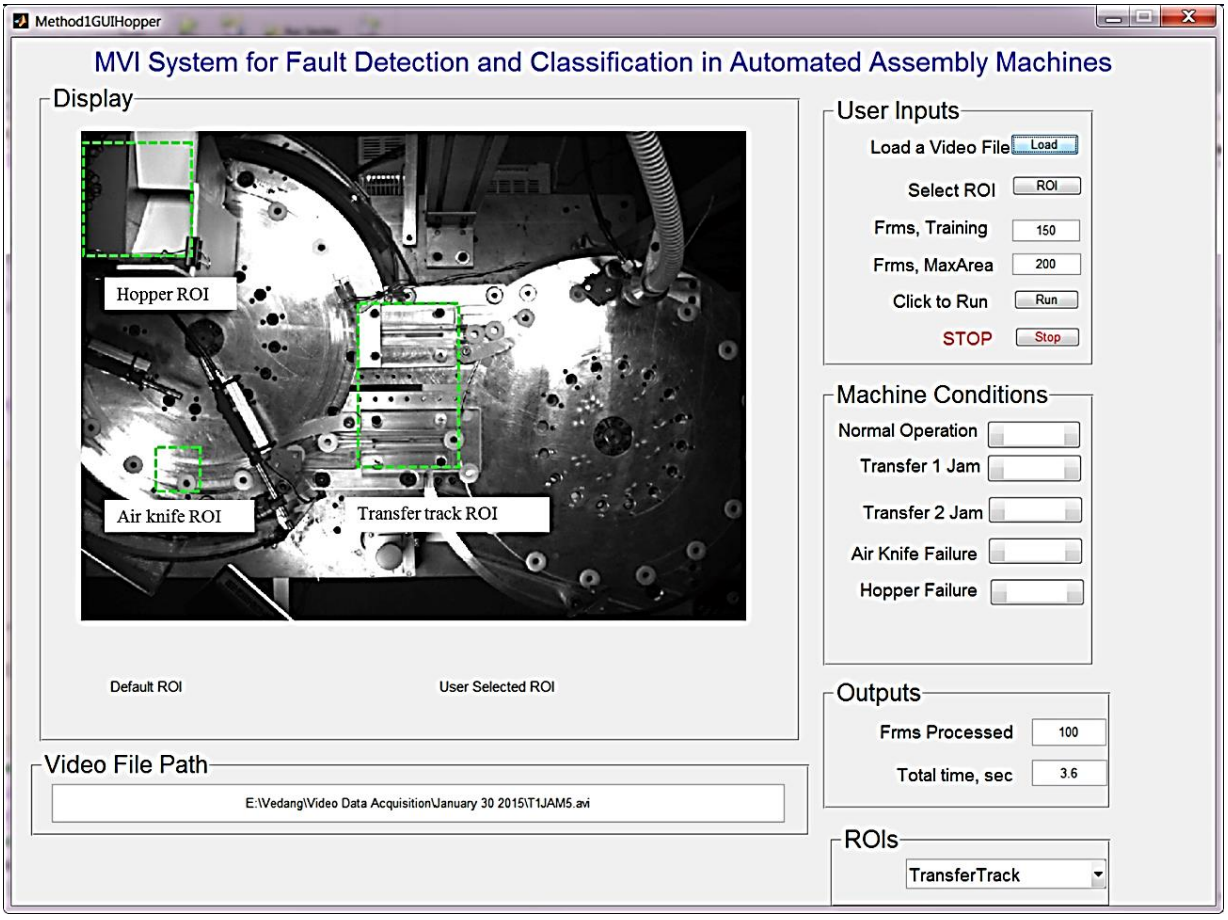


Figure 4-2 Fault detection GUI designed in MATLAB

4.2 Video Data Acquisition

Video data were acquired for both normal and faulty operating conditions of the machine with an overhead camera. During the setup of the MVI system, the correct height of the camera was determined as explained in Chapter 3. The camera was mounted 1.1 m above the surface of the machine in order to cover the entire machine surface within the Field of View (FOV) of the camera. The machine was run for normal operation and five sets of videos were recorded. The faults were introduced by the HMI-PLC and videos were recorded for each fault condition. During video acquisition of the data for the faults, the machine was first run under normal operation for up 200 frames and then the fault was introduced. The 200 frames recorded one full rotation of the primary and secondary wheels. The cycle of assembly events was repeated after one full rotation. The frame number at the event of the fault was recorded. The video data sets were acquired with LabVIEW and IEEE 1394 camera during the preliminary phase of testing. Only the transfer track faults were recorded during this phase. For the formal set of the experiments, after the expansion of the ROIs, a high speed USB 3.0 camera with MATLAB was used for the video data acquisition. The next two subsections describe the video acquisition procedure in more detail.

4.2.1 Acquisition with an IEEE Camera and LabVIEW

The first step in video data acquisition was the selection of a proper camera and lights. The next step was to mount the camera at an appropriate height such that the potential fault locations could be seen by the camera. These steps were performed as per the procedure given in Chapter 3. Initially, an IMI TECH IEEE1394 digital camera was used for video data acquisition. The camera, as shown in Figure 4-3, had a CCD sensor with a 1/3" sensor size and video data acquisition rate of 30 fps (frames per second). Two LED panel lights with diffusers were setup on both sides of the machine near the transfer track region. One of the major challenges in video data acquisition was found to be the reflection of the light from the surfaces of the polished wheels. This issue was solved by positioning the LED lights in such a way that a dark field lighting effect was obtained. The advantage of a dark field light is that it avoids the specular reflections and highlights only the required details on a shiny background. The position of the lights and camera is shown in Figure 4-4.

Figure 4-3 IMI Tech IEEE1394 digital camera with lens

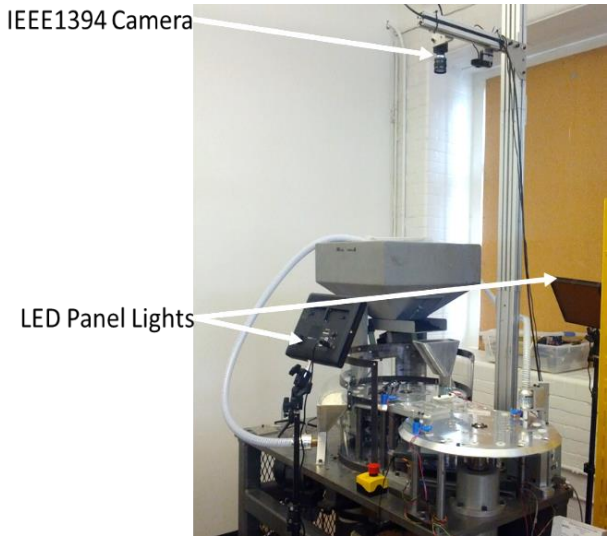


Figure 4-4 MVI with IEEE 1394 digital camera and two-LED panel lights

LabVIEW 2010 software from National Instruments Corporation was used for the preliminary video data acquisition with the IEEE 1394 camera. Figure 4-5 shows the front panel LabVIEW GUI for video data acquisition. The required user inputs were frames per second and the number of frames to be captured. The video was stored in Audio Video Interleave (AVI) format at a user selected path. The stop button was used to stop the data acquisition at any time on a basis as needed.

The LabVIEW block diagram for the acquisition program is shown in Figure 4-6. LabVIEW is a system design platform and development environment that uses a graphical programming language. Hence, the block diagram represents the graphical program for video data acquisition in LabVIEW.

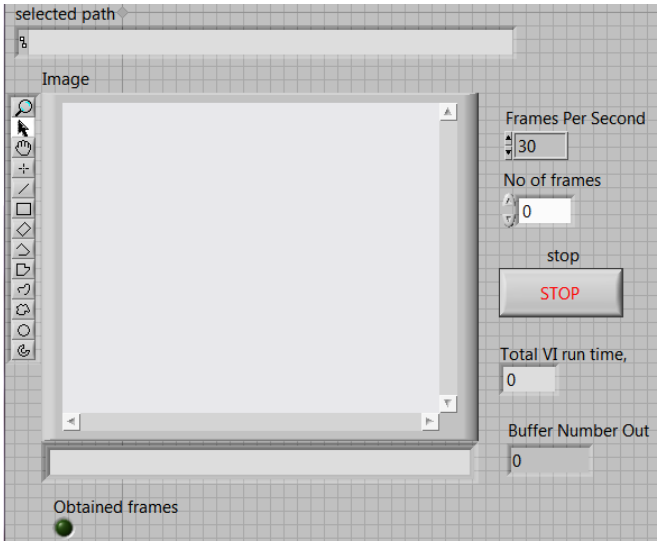


Figure 4-5 LabVIEW front panel for video data acquisition

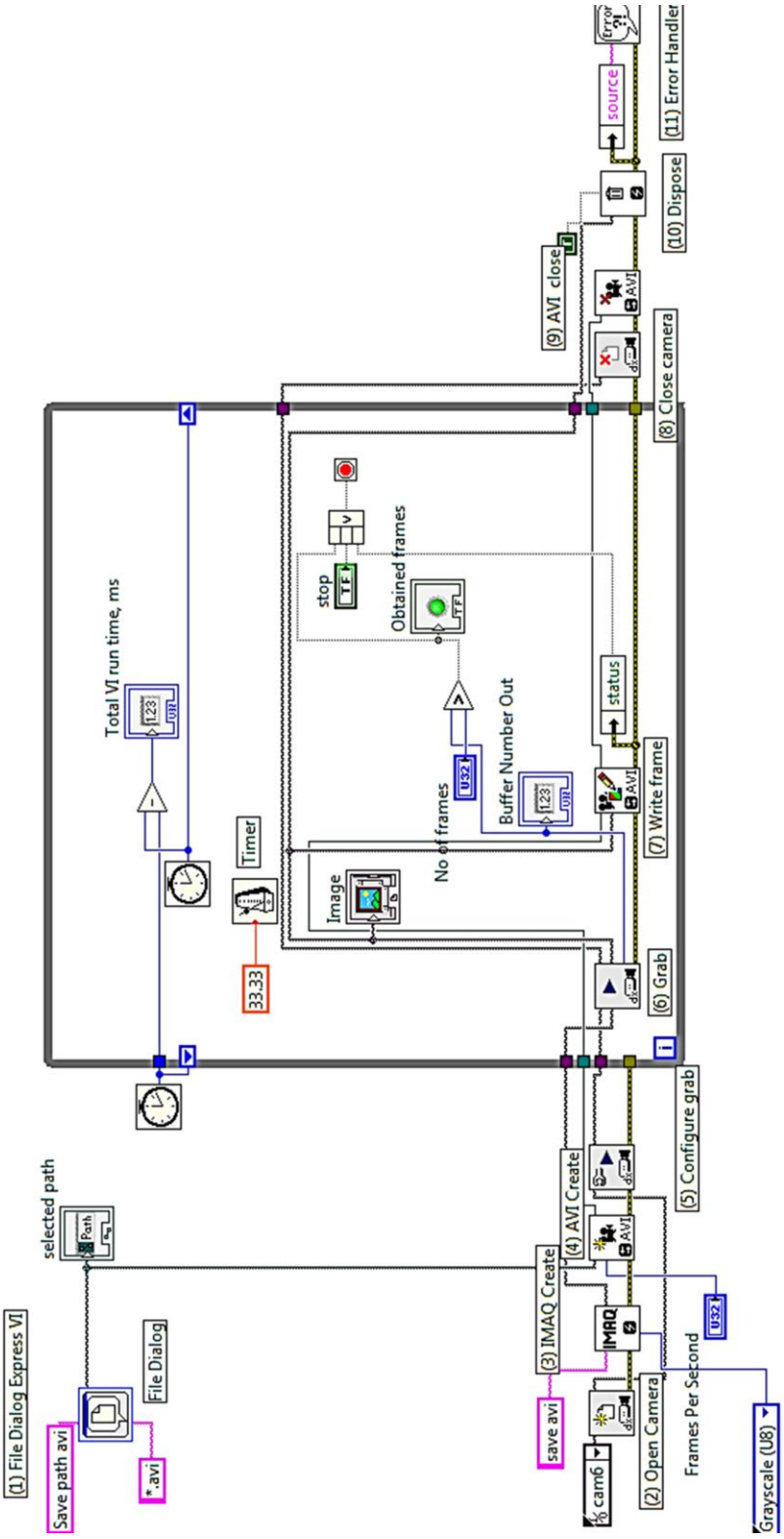


Figure 4-6 LabVIEW block diagram for video data acquisition

The main steps in the program were initialization of the camera, set up of the AVI file and the file path, acquiring image frames and storing them in the buffer, writing the frames to the AVI file until required number of frames were obtained. The main blocks and their functions in Figure 4-6 are explained below (LabVIEW, 2010). The numbers line up with the numbers in the figure.

(1) File Dialog Express : It displays a dialog box with which a user can specify the path to a file or directory. The video data is saved to that file.

(2) Open camera: It opens a camera, queries the camera for its capabilities, load a camera configuration file and creates a unique reference to the camera. In the program, the camera was cam6.

(3) IMAQ Create: Creates a temporary memory location for an image.

(4) AVI (Audio Video Interleave) Create: Creates a new AVI or rewrites an old AVI file. It outputs a unique reference number for the AVI file.

(5) Configure grab: Configures and starts a grab acquisition.

(6) Grab: Acquires the most recent frame and outputs it to the next block. This block runs in a loop.

(7) Write frame: Writes an image to the AVI file specified by AVI reference number. This block runs in a loop.

(8) Close camera: Stops an acquistion in progress, releases resourses associated with an acquisition and closes the specified camera session.

(9) AVI close: Closes the specified AVI file.

(10) Dispose: Destroys an image and frees the space it has occupied in memory. Cleans the buffer.

(11) Error handler: Checks for an error and if error occurs displays the information about the error.

(12) Timers and counters: The timer block controls the loop execution rate. To acquire an AVI with 30 frames per second, it executes all blocks in the loop at a rate of 33.33 ms per loop. The counter keeps track of the number of frame acquired and it generates the stop signal when the number matches the user specified number of frames.

After preliminary testing with the videos acquired with the IEEE 1394 camera, it was found it was hard to illuminate the overall machine surface with only just two lights. The IEEE 1394 camera had low bandwidth and the videos were of low quality due to the small (1/3”) image sensor. Hence, there was a need for a camera with a better sensor and higher bandwidth. Also, two more LED lights were added to illuminate

the required machine surface area properly. Therefore, a USB 3.0 camera and 4 LED lights were used for all the video acquisition during the formal set of experiments. Since, MATLAB could be more easily connected to the USB 3.0 camera than LabVIEW and it has Computer Vision Tool box for video data processing, MATLAB was used as the software for the processing of images from the USB 3.0 camera.

4.2.2 Acquisition with a USB Camera and MATLAB

The IEEE 1394 camera was replaced with a USB 3.0 camera for the following reasons:

1. USB 3.0 is a standard hardware interface
2. High bandwidth: The USB 3.0 camera can support up to 350 MB/s while IEEE 1394a has max 32 MB/s and IEEE 1394b has max 64 MB/s bandwidths.
3. Easy to plug and play.
4. USB 3.0 has low CPU load, single latency and jitter.
5. Low power consumption and energy management option that can be used to set a device to a suspend mode when it is not required
6. One cable solution. That means power via USB 3.0 cable.
7. USB 3.0 camera can handle a cable length upto 8 meters that was sufficient based on the project requirements.

The specific USB 3.0 camera selected was a Grasshopper3 2.8 MP Color USB 3.0 digital camera with a Sony IX687, 1/1.8" sensor. The camera has various acquisition modes and resolution options. The camera setting can be configured and controlled by Point Grey's FlyCapture software. The FlyCapture Software Development Kit (SDK) provides a common software interface to control and acquire images with Point Grey USB 3.0, GigE, FireWire, and USB 2.0 cameras with the same API under 32- or 64-bit Windows or Linux. FlyCapture supports ActiveX and DirectShow interfaces, and includes the FirePRO low-level 1394b interface driver, enhanced USB 3.0 interface driver, and the GigE image filter driver. It has a complete software API library, ready-to-use demo programs, and comprehensive source code examples. Figure 4-7 shows the Flycap software main window labelled with the functions of the various buttons.

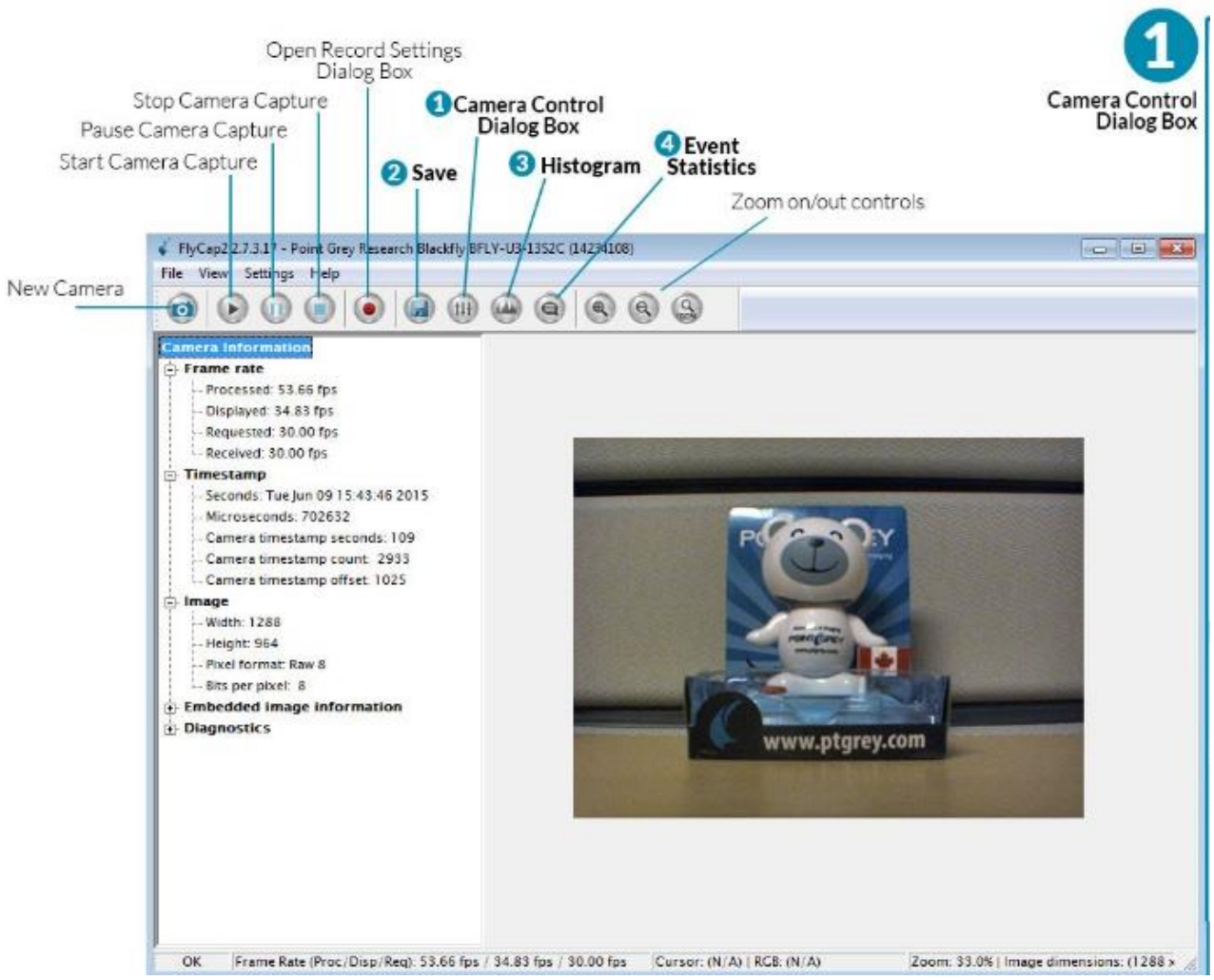


Figure 4-7 Flycap2 software main window (image source: (FlyCaptureSDK, 2015))

The Grasshopper USB 3.0 camera was configured with the flycap2 software and the camera's control parameters such as brightness, sharpness, gain and hue were set with the software. The frame rate was set at 30 fps and the shutter and gain parameters were set to automatic mode to compensate for light variations. These parameters controlled the quality of an image under minor fluctuations in the lighting conditions. Figure 4-8 shows the camera settings with the flycap2 software. The camera had some standards modes and an option to set a custom mode. A mode is a combination of camera resolution and fps. The desired resolution and acquisition rate combination was not available in any of the standard modes. Hence, the camera mode was set to custom mode 4 that had pixel format mono 8, camera resolution of 964 x 724 and video data acquisition rate of 30 fps. The adopted camera mode settings are shown in Figure 4-9.

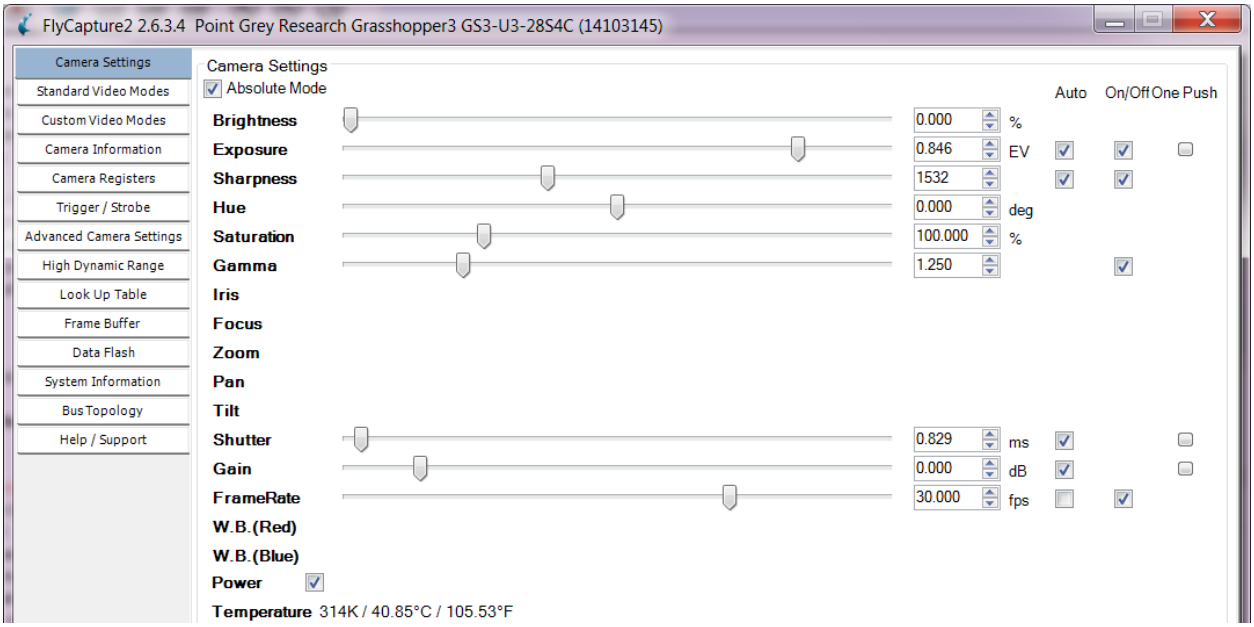


Figure 4-8 Grasshopper3 USB 3.0 camera settings in Flycap2 software

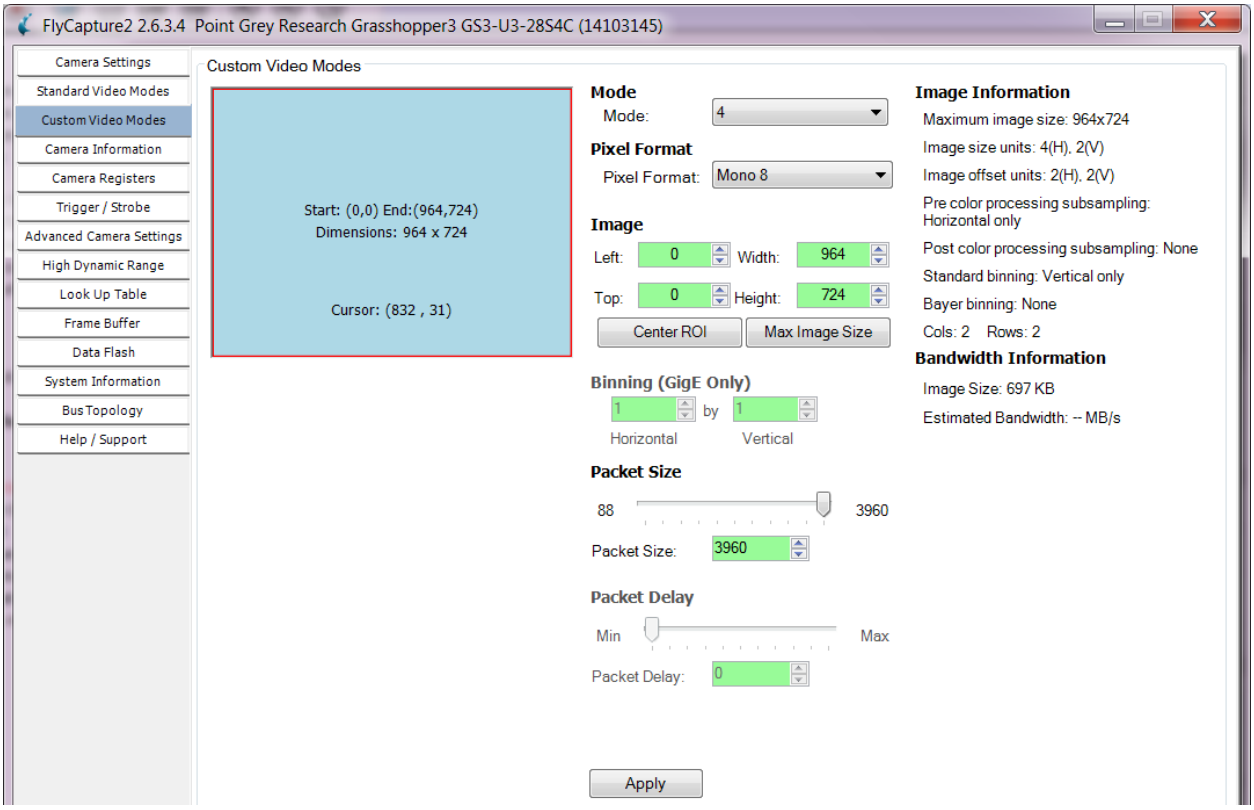


Figure 4-9 Grasshopper3 USB 3.0 camera mode and image resolution in Flycap2 software

Once the camera parameters were set, a GUI designed in MATLAB as shown in Figure 4-10, was used for video data acquisition. The user is required to input camera details and the required amount of video acquisition time. The user could also select the ROI from the FOV of the camera. The frames were acquired at the specified fps and the video file was created by combining the frames. The video file was stored in AVI format at the location specified by the user. The MATLAB code information is provided in Appendix C.

The video data acquisition was performed for both normal and faulty operation of the machine. For each condition, five videos data sets were recorded. Both normal and fault condition videos were recorded for each of the faults (transfer track 1 and 2, air knife and hopper). Therefore, a total of 7 distinct types of operating conditions were recorded as shown in Table 4-1. Three videos were acquired for normal operation for each ROI and four videos were acquired for each ROI under faulty operation. For better representation of the machine's operating conditions, five videos were acquired per condition. As a result, a total 35 (i.e. 7 times 5) videos were acquired. All video files were processed by each fault detection method. Each video file had an average of 600 frames that resulted in around 3000 frames (images) for one operating condition of the machine. Hence, 35 video files were enough for training and testing with the MVI methods. Xiaokun and Porikli (2004) used 12 video files to detect and classify highway traffic events into six categories. Szkilnyk (2012) used 20 video files of faulty operation sequence for fault detection and classification using video event detection method based on STVs.

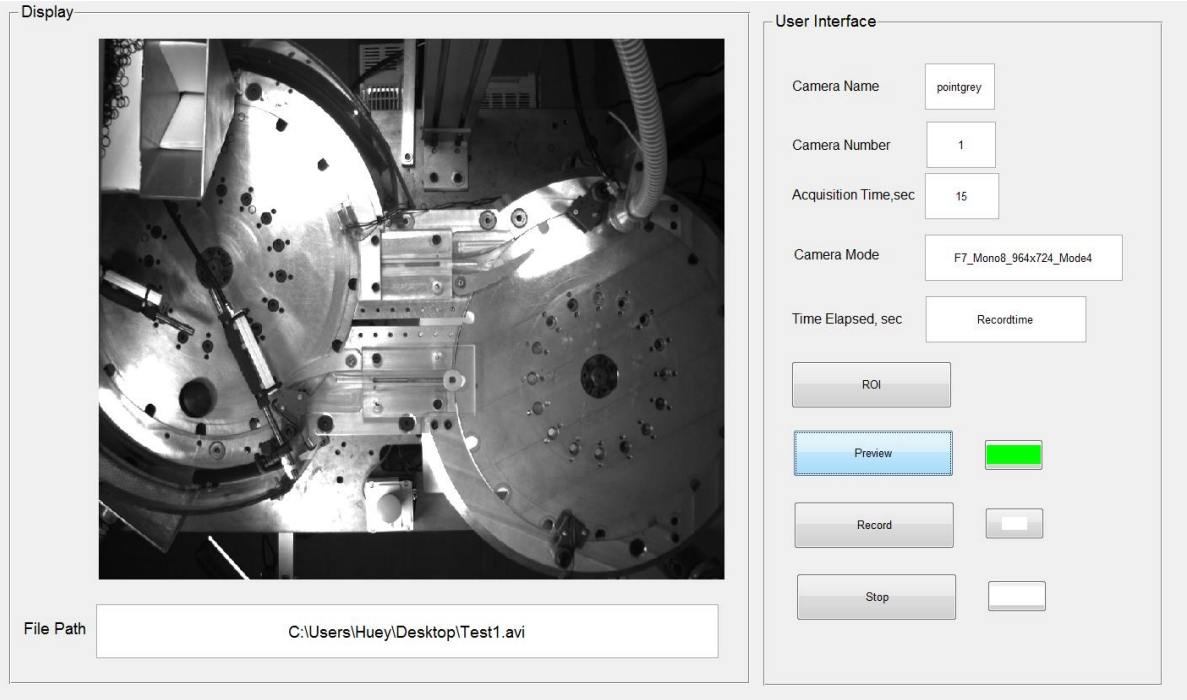


Figure 4-10 MATLAB GUI for video data acquisition

Table 4-1 Summary of video data set

ROI	Condition	Number of video files
Transfer track	Normal operation	5
Transfer track	Transfer 1 jam (Fault F1)	5
Transfer track	Transfer 2 jam (Fault F2)	5
Air Knife	Normal operation	5
Air Knife	Air Knife Fault (Fault F3)	5
Hopper	Normal operation	5
Hopper	Hopper Fault (Fault F4)	5
Total		35

The faults were introduced at some point during video acquisition with the HMI-PLC. The frame number at which the fault was introduced (by pressing a button on the HMI) was recorded for each operating condition. Table 4-2 lists all the video files with the total number of frames in the video files. For faulty operation, the frame number at what time a fault was introduced was also recorded in the table. The transfer track region has 15 video files: 5 video files for a normal operation through the transfer track region (NO 1 to 5), 5 video files for transfer track 1 jam (T1JAM 1 to 5) and 5 video files for transfer track 2 jam (T2 JAM 1 to 5). The air knife fault has 10 video files: 5 video files for a normal operation through the air knife region (AirNO 1 to 5) and 5 video files for the air knife fault (Air knife fault 1 to 5). Similarly, the hopper region has 10 video files: 5 video files for a normal operation through the hopper region (Hopper NO 1 to 5) and 5 video files for the hopper fault (Hopper fault 1 to 5). For video data acquisition with faults, the machine was run under a normal operation for an average of 200 frames (1 full rotation of the wheels) and then a fault was introduced.

Table 4-2 Video data frame nos. for normal operation and frame nos. at fault introduction

Sr. No.	Video data file	Total no. of frames	Frame no. at fault introduction
Transfer Track Region			
1	NO1	582	
2	NO2	627	
3	NO3	627	
4	NO4	627	
5	NO5	584	
6	T1JAM 1	546	300
7	T1JAM 2	545	300
8	T1JAM 3	499	248
9	T1JAM 4	544	289
10	T1JAM 5	544	297
11	T2JAM 1	500	252
12	T2JAM 2	628	390
13	T2JAM 3	328	320
14	T2JAM 4	629	342
15	T2JAM 5	628	357
Air Knife Region			
18	AirNO 1	503	
19	AirNO 2	501	
18	AirNO 3	503	
19	AirNO 4	501	
20	AirNO 5	503	
21	Air knife fault 1	539	290
22	Air knife fault 2	627	275
23	Air knife fault 3	671	353
24	Air knife fault 4	587	205
25	Air knife fault 5	623	238
Hopper Region			
26	Hopper NO 1	623	
27	Hopper NO 2	609	
28	Hopper NO 3	610	
29	Hopper NO 4	549	
30	Hopper NO 5	561	
31	Hopper fault 1	547	381
32	Hopper fault 2	560	230
33	Hopper fault 3	609	330
34	Hopper fault 4	623	320
35	Hopper fault 5	609	307

4.3 Method 1: Fault Detection with GMMs and Blob Analysis

Method 1 uses the blob area of foreground pixels as a feature to detect and classify the operating condition of the O-ring machine. The blob area is the area of the connected pixels in the estimated foreground. The foreground estimation is performed by means of a mixture of Gaussians. The method learns a normal operation sequence by calculating and monitoring the blob area for the first few initial frames. The blob area for the normal operation depends on the number of moving objects in the foreground and it should be within some limits. This value is considered as a reference value. If the blob area for any frame exceeds the reference value estimated during the training phase, then it would be an indication of a jam. Similarly, if the blob area for any frame drops below its estimated reference value then it would be an indication of a missing part. Whenever the deviation is detected for the blob area with respected to its reference value, the method keeps track of the blob area for the next few consecutive frames. If the blob area for the immediate frame returns within its limits, then that is the case of a temporary fault and the method does not generate any warning for an operator. However, if the blob area exceeds the limit and stays outside the limits for next few consecutive frames, the method considers that as a fault, sets the fault indicator red and warns an operator to take immediate action. Section 4.3.1 explains the Gaussian Mixture Model approach for foreground estimation. Section 4.3.2 describes blob analysis with connected components and Section 4.3.3 covers algorithm development for Method 1.

4.3.1 Foreground Estimation with Mixture of Gaussians

This method uses the foreground detector system object to monitor the machine condition while it is in operation and classifies the operating condition into a normal operation or a faulty operation. The foreground detector object reads grayscale video frames to compute and return the foreground mask with Gaussian Mixture Models (GMMs) (Stauffer and Grimson, 1999 and Kaewtrakulpong and Bowden, 2001).

GMMs Concept

GMMs is a clustering technique that is based on a probability density estimation and Expectation-Maximization (EM) to fit the model parameters that can classify a feature space into a set of overlapping clusters. It is the advanced form of K-means algorithm for clustering of data points in the feature space. K-means describes each cluster center with just a single point in the feature space and assigns data points to its nearest clusters. K-means cannot give reliable and accurate representation of the clusters if: (a) clusters are overlapping or (b) clusters are non-circular in shape (McLachlan and Peel, 2004). In case (a), it is hard to say which assignment is right since there is more than one possibility. In case (b), as K-means uses the Euclidean distance to the centers, it cannot handle non-circular cluster shapes. The ambiguity, for case (a)

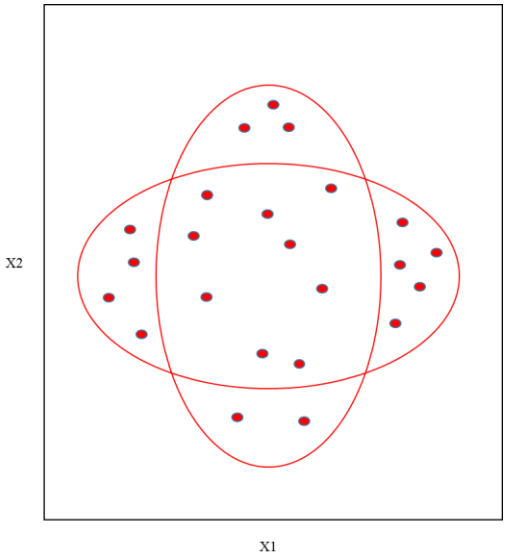


Figure 4-11 Overlapping clusters in a two dimensional space

and (b), is shown in Figure 4-11. The data in the figure can be seen forming two groups, both are centered at the same place but one has more spread in X1 and the other has more spread in X2.

It is visible that the clusters are overlapping, but K-means will not be able to discover that assignment. The problem can be solved by GMMs which are an extension to K-means in which clusters are modeled with Gaussian distributions. A cluster in GMMs is represented not only by its mean but also a covariance that describes its ellipsoidal shape. The model can be fit to the data by maximizing likelihood for the observed data. The likelihood is maximized using Expectation-Maximization (EM) that is very reminiscent of K-means. EM assigns each data to a cluster using some soft probability. Once GMMs completes the clustering, it creates a generative model for the data. The model can be used for sampling new examples that are similar to the measured data. It can also be used to compare two data sets (e.g., training and testing) to see if they differ. As a third application it can be used to infer missing data values from some of the measured data. Once a GMMs learns the training data, it can assign each sample of the test data to the class of Gaussian it belongs to with a probability value.

GMMs uses mean μ_c, variance σ_c and size π_c for each cluster to calculate a joint probability distribution using Equation (4.1). It is the weighted average of individual components.

$$P(x) = \sum_c \pi_c \times \eta(x; \mu_c, \sigma_c) \tag{4.1}$$

For high dimensional multivariate features, the distribution is given by Equation (4.2), where $\vec{X}$ and $\vec{\mu}$ are vectors and n is the number of features in data point X:

$$\eta\left(\vec{X},\vec{\mu},\sum\right)=\frac{1}{2\pi^{\frac{n}{2}}|\sum|^{\frac{1}{2}}}e^{-\frac{1}{2}\left(\vec{X}-\vec{\mu}\right)^{T}\sum^{-1}(\vec{x}-\vec{\mu})} \tag{4.2}$$

Maximum likelihood for the multivariate Gaussian model are mean (1st moment of the data) and covariance (centered 2nd moment of the data) estimates. EM then proceeds iteratively in two steps: E-step (Expectation) and M-step (Maximization).

E-step treats the Gaussian parameters mean μ_c, covariance Σ_c and size π_c as fixed. For each data point i in each cluster c, its relative probability that it belongs to cluster c is calculated and normalized by dividing with the total overall values for c using the following equation:

$$r_{i,c}=\frac{\pi_c\times\eta(x_i;\mu_c,\sum_c\quad)}{\sum_{c'}\pi_{c'}\times\eta(x_i;\mu_{c'},\sum_{c'}\quad)} \tag{4.3}$$

In M-step, for each cluster c, the assigned probabilities $r_{i,c}$ are fixed and the parameters of the cluster mean μ_c, covariance Σc and size π_c are updated using Equations (4.4) to (4.7).

Total responsibility allotted to cluster c,

$$m_c=\sum_i r_{i,c} \tag{4.4}$$

Fraction of total assigned to cluster c,

$$\pi_c=\frac{m_c}{m} \tag{4.5}$$

Weighted mean of the assigned data,

$$\mu_c=\frac{1}{m_c}\sum_i r_{i,c}x^{(i)} \tag{4.6}$$

Weighted covariance of the assigned data calculated using new means,

$$\sum_c\quad=\frac{1}{m_c}\sum_i r_{i,c}\left(x^{(i)}-\mu_c\right)^T\left(x^{(i)}-\mu_c\right) \tag{4.7}$$

These iterations increase log-likelihood of the data. Log-likelihood is the log probability of the data under mixture model given by Equation (4.8).

$$\log p(X)=\sum_i\log\left[\sum_c\pi_c\times\eta\left(x_i;\mu_c,\sum_c\quad\right)\right] \tag{4.8}$$

The equations are iterated until convergence is obtained using the ascent method. The convergence is slow and not as abrupt as with K-means. Also, convergence is not guaranteed to be a global minimum, hence several initializations should be used and log-likelihood decides the best initialization.

GMMs for Foreground Estimation

Foreground is estimated by subtracting background from a frame. Background detection is the first step for foreground estimation. Simple background models do not give proper estimates of a background in cases such as: lighting changes, repetitive motions from clutter that result in the presence of different objects at the same pixel location (i, j) and long-term scene changes. As a result of such conditions, changes in the background appear at a rate faster than that of the background model update rate. In these cases, a single Gaussian per pixel is not an adequate for the background model.

To overcome the limitations of a single Gaussian per pixel in the background model, Stauffer & Grimson (1999) presented a mixture of adaptive Gaussians per pixel as the background model. The model provides a description of both foreground and background pixels. Hence, the approach of an adaptive Gaussian mixture model, where each pixel in the scene is modeled by a mixture of K Gaussian distributions works better for moving object detection. The approach compares a video frame to a background model to determine whether individual pixels are part of the background or the foreground. Based on this information, it computes the foreground mask. Once computed, it performs background subtraction and detects foreground objects in an image taken from the overhead stationary camera. The three steps for background detection are: construction of a GMMs, parameter update and background estimation.

Construction of GMMs

If I is the image sequence and $\{x_0, y_0\}$ represent the location of a pixel, then the history of the pixel value X_t is obtained by equation (4.9),

$$\{X_1, \ldots\ldots, X_t\} = \{I(x_0, y_0, i): 1 \leq i \leq t\} \quad (4.9)$$

The recent history of each pixel is modelled by a mixture of K Gaussian distributions. In most cases, K is set to be between 3 and 5. The probability of observing a certain pixel value X_t at time t can be written as Equation (4.10), where $\omega_{i,t}$ is an estimate of the weight for ith Gaussian component, $\mu_{i,t}$ is the mean of the Gaussian component and $\eta(X_t, \mu, \sum)$ is the Gaussian probability density function (Equation 4.11). To simplify computations, the co-variance matrix is assumed to be diagonal and of the form $\sum_i = \sigma_i^2 I$. This assumes that the red, green and blue pixel values are independent and have the same variances.

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \times \eta(X_t, \mu_{i,t}, \sum_{i,t}) \tag{4.10}$$

$$\eta(X_t, \mu, \sum) = \frac{1}{2\pi^{\frac{n}{2}} |\sum|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_t)^T \sum_k^{-1} (x-\mu_k)} \tag{4.11}$$

Using the history of pixels from a set of frames, each pixel value in the scene is characterized by a mixture of Gaussians. If the history is considered as stationary, the classification of foreground and background clusters can be obtained by maximizing the likelihood of the observed data using EM. The history of pixel in real application is not stationary and it varies with time t. Hence, at each frame at a time t, two problems must be solved simultaneously: 1) assignment of new observed values to the best matching distribution and 2) estimating the updated model parameters. The parameters are updated using online K-means approximation.

Parameters Update

Each new pixel value is compared with the existing K Gaussian distributions, until a match is found. If a pixel value is within 2.5 standard deviations of a distribution, then it is considered as a match. Once the match is found the parameters are updated by the following equations (4.12) to (4.15).

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha \times M_{k,t} \tag{4.12}$$

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho \times X_t \tag{4.13}$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T (X_t - \mu_t) \tag{4.14}$$

$$\rho = \alpha \times \eta(X_t | \mu_k, \sigma_k) \tag{4.15}$$

where, α is the learning rate (range 0 to 1), ρ is the second learning rate, $M_{k,t}$ is 1 for the model that matched and 0 for other models. The μ and σ parameters are updated only when the current pixel value matches one of the distributions. In case of no match found, the distribution is replaced with a new distribution using the current value of the pixel as its mean, high initial variance and low prior weight.

Background Estimation

The K distributions are ranked as per their fitness value which is the ratio of the peak amplitude to the standard deviation, ω_k/σ_k. The variance of the moving object is expected to stay larger than the background pixel. Hence, it is assumed that the background has the higher and more compact distributions. As a result of ranking, a new distribution is generated with the most likely distributions on top and the less probable ones on the bottom. The estimated background B is used to compute the foreground mask:

$$B = \arg\min_{\mathrm{b}} \left(\sum_{k=1}^{b} \omega_k > T\right) \tag{4.16}$$

The first b distributions, in ranking order, are used for modelling the background, where T is a threshold which is the measure of the minimum portion of the data that should be accounted for by the background model. The estimated output from a frame is a set of pixels having higher probability that belong to the foreground. The foreground mask contains pixels that represent both the actual moving objects and noise due to random motions. Smoothing spatial filters are used on the estimated foreground for noise reduction before the blob analysis.

4.3.2 Foreground Blob Analysis

Blob analysis is a fundamental image processing technique based on the analysis of consistent image regions. It groups image pixels into a set of regions where each region has connected pixels within it. The first step for blob analysis is image segmentation. Segmentation is a process of partitioning a digital image into multiple sets of pixels. Pixels in one set are similar with respect to some features. Segmentation isolates objects in images with respect to other objects and a background. Segmentation of an image can be performed by two approaches: boundary detection and region growing. Boundary-based segmentation creates partitions in an image based on sudden changes in intensity, such as edges in the image. It enhances the boundary pixels of objects within the image and subsequently isolates them from the rest of the image elements. It extracts the overall shape of objects from the image. First or second order image derivations are used for boundary-based segmentation. Region-based segmentation is a similarity based approach that creates partitions by grouping pixels into regions that are similar according to a set of predefined criteria. Thresholding is an example of the similarity based segmentation (Gonzalez and Woods, 2002).

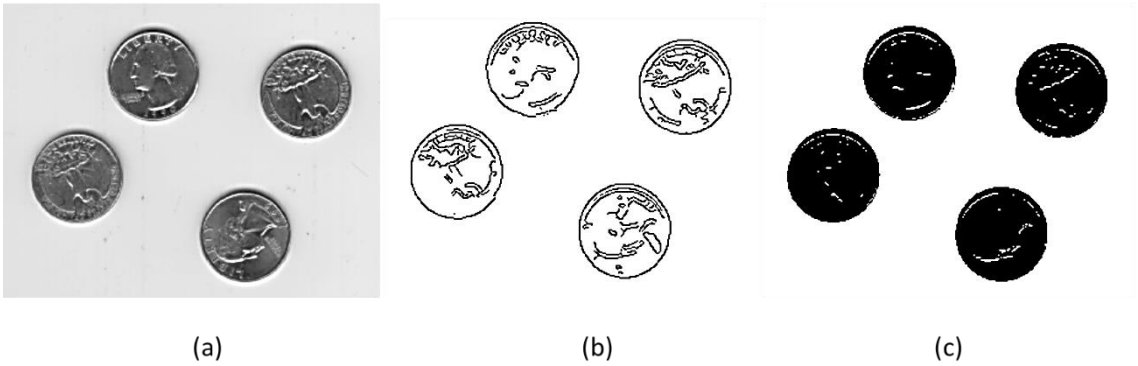


Figure 4-12 Segmentation examples: (a) original image, (b) boundary-based segmentation and (c) region-based segmentation (image source: MATLAB, 2014b)

Segmentation of coin images with both approaches is shown in Figure 4-12. The original coin images have a dark appearance with a gray background. The second set of images is obtained by edge detection with a "canny approach" (boundary-based). The third set of images is the result of segmentation using a global thresholding approach (region-based). The result of segmentation is a binary image (only two colors) with a different color for the foreground (i.e. objects) and the background (i.e. slow changes in intensity). The next step after segmentation is a connectivity check, where the connected pixels are considered as an object. For a pixel at location (x,y), the center, all pixels that are at locations $\{(x+1,y), (x-1, y), (x, y+1), (x, y-1)\}$ are called 4-neighbors, N_4. The pixels that are at locations $\{(x+1,y+1),(x-1, y+1), (x+1, y-1), (x-1, y-1)\}$ are called diagonal neighbors, N_D , for the center pixel. The pixels that combine diagonal neighbors with 4-neighbors are called 8-neighbors of p. Pixel neighbors are shown in Figure 4-13.

Two pixels are said to be connected if they are neighbors and their gray level intensities satisfy a specified criterion of similarity (for a binary image, their intensity value is the same). The pixels are considered to be 4-connected if they are 4-neighbors and their gray levels are similar. Similarly, if pixels are 8-neighbors and their gray levels are similar, they are considered as 8-connected. Connectivity analysis of a segmented image results in the blobs of connected pixels. Depending on the definition used for connectivity, the size and number of the blobs can be different for the same segmented image.

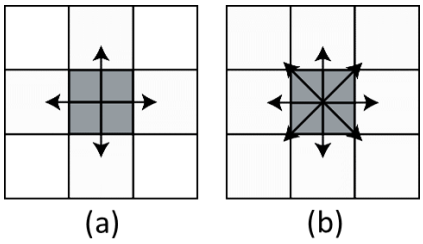


Figure 4-13 Pixel neighbors: (a) 4-neighbors, (b) 8-neighbors (image source: MATLAB, 2014b)

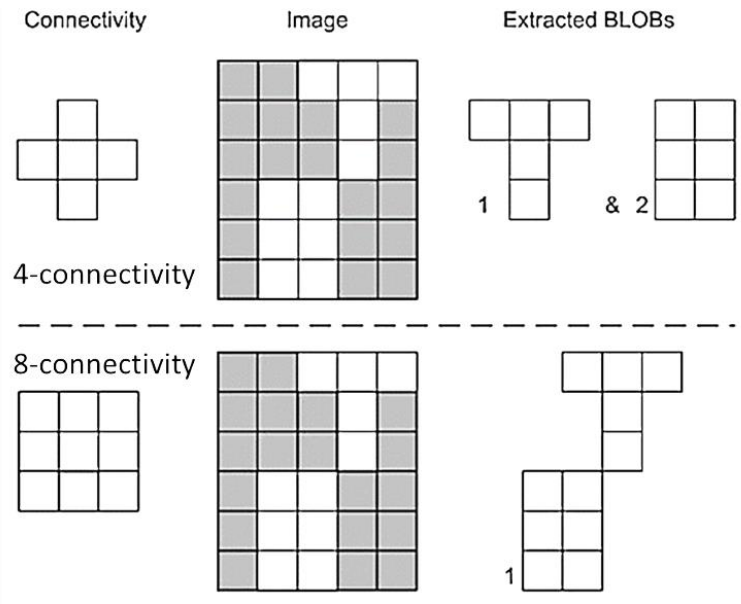


Figure 4-14 Result of connectivity analysis: upper figures used 4-connectivity, lower figures used 8-connectivity (image source: Blob Analysis, 2015)

Figure 4-14 shows the result of two different connectivity used for connected component analysis. White pixels are foreground and black pixels are background in the image. Using 4-connectivity results in 2 blobs, while 8-connectivity results in a single blob. 4-connectivity requires fewer computations and it can process the image faster, but 8-connectivity is more accurate than 4-connectivity.

The blob analysis computes statistics for the connected regions in a binary segmented image. It is a way of representing each blob by a number of characteristics such as centroid, area of the blob, bounding box, perimeter, major and minor axes lengths for ellipses and label. The definitions for each of the characteristics are given below.

Centroid: It is the (*x*,*y*) location of the centroid of a single blob. For more than one blobs, it is M x 2 matrix of centroid coordinates, where M represents the number of blobs.

Blob Area: It represents number of pixels in each blob. For more than one blobs, it is a vector of size M, where M is the number of blobs.

Bounding Box: It represents the bonding box coordinates as a set {*x, y, width, height*} for each blob, where (*x, y*) indicates the upper left corner of the bounding box. For M blobs, it will be M x 4 matrix.

Perimeter: It is the measure of a perimeter length in pixels for each blob. For M blobs, it will be a vector of length M.

Major and Minor Axes: They represent the lengths of major and minor axes for an ellipse. For M ellipse, each will be a vector of size M.

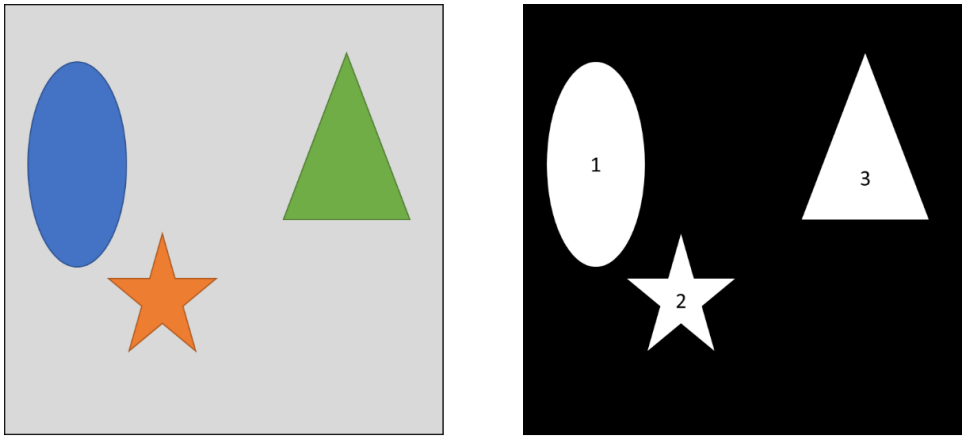


Figure 4-15 Blob extraction and analysis: (a) original image (b) binary image with blob labels

Label: Each blob is assigned a unique label. The label matrix is of same size that of the image with label number at corresponding pixel locations.

Figure 4-15 illustrates blob extraction and analysis. Figure 4-15(a) is the original image with 3 different shapes: ellipse, star and triangle. Figure 4-15(b) is the result of region-based segmentation that shows three labels: 1, 2 and 3 assigned to the different objects. The connected component analysis considers all the white pixels in the ellipse as one object. Similarly, all the white pixels in the star and in the triangle make other isolated objects. Blob analysis represents each blob using a set of numbers calculated from different features (MATLAB, 2014b).

The GMMs based fault detection method extracts the foreground from each frame, performs segmentation and connected component labelling. It then applies blob extraction and analysis to get the area and bounding box information for all the blobs in each frame. These are the features used to classify the machine condition into a normal or faulty operation using the algorithm explained in the next section.

4.3.3 Implementation of Method 1

The application of Method 1 to the problem at hand was not straight forward. This section describes the technique that was developed to enable implementation. The implementation for Method 1 is given as steps in Table 4-3 and as a flowchart in Figure 4-16. The MATLAB code information is provided in Appendix C. The steps are explained assuming normal operation through the transfer track region.

The first step is to create system objects to read a video file, estimate a foreground with GMMs and then to perform blob analysis. The video is read frame by frame and each frame is cropped to the ROI defined by the user. For the transfer track region, the full image and the cropped ROI are shown in Figure 4-17 and Figure 4-18, respectively.

Table 4-3 Steps for implementation of Method 1

Stage	Step No.	Description
Fault Detection	1	Clear workspace variables
	2	Create system objects *(file read, player, foreground detector, blob analysis)*
	3	Initialize variables *(No. of frames for training, no. of frames to estimate maximum area, ROI)*
	4	Read a frame at a time, after the last frame read go to ***step 16***
	5	Crop ROI and detect foreground with GMM
	6	Perform blob analysis of detected objects
	7	Compute an area, a bounding box and no. of frames having zero area
	8	Output a frame to the player
	9	Is no. of frames greater than no. of frames for training? If No, go to ***step 4***. If yes, continue to next step
	10	Calculate the maximum blob area (threshold)
	11	(Is last 5 frames blob area greater than the maximum blob area) OR (no. of frames having zero blob area is greater than the no. of frames with zero blob area during the training phase)? If no, go to ***step 12.*** If yes, go to ***step 13***
Fault Classification	12	Set normal operation LED on and go to ***step 4***
	13	Is a bounding box in lower part of the ROI? If yes, go to ***step 14*** else go to ***step 15***
	14	Set transfer 1 jam LED on, stop processing and go to ***step 16***
	15	Set transfer 2 jam LED on, stop processing and go to ***step 16***
	16	Release all objects from the memory

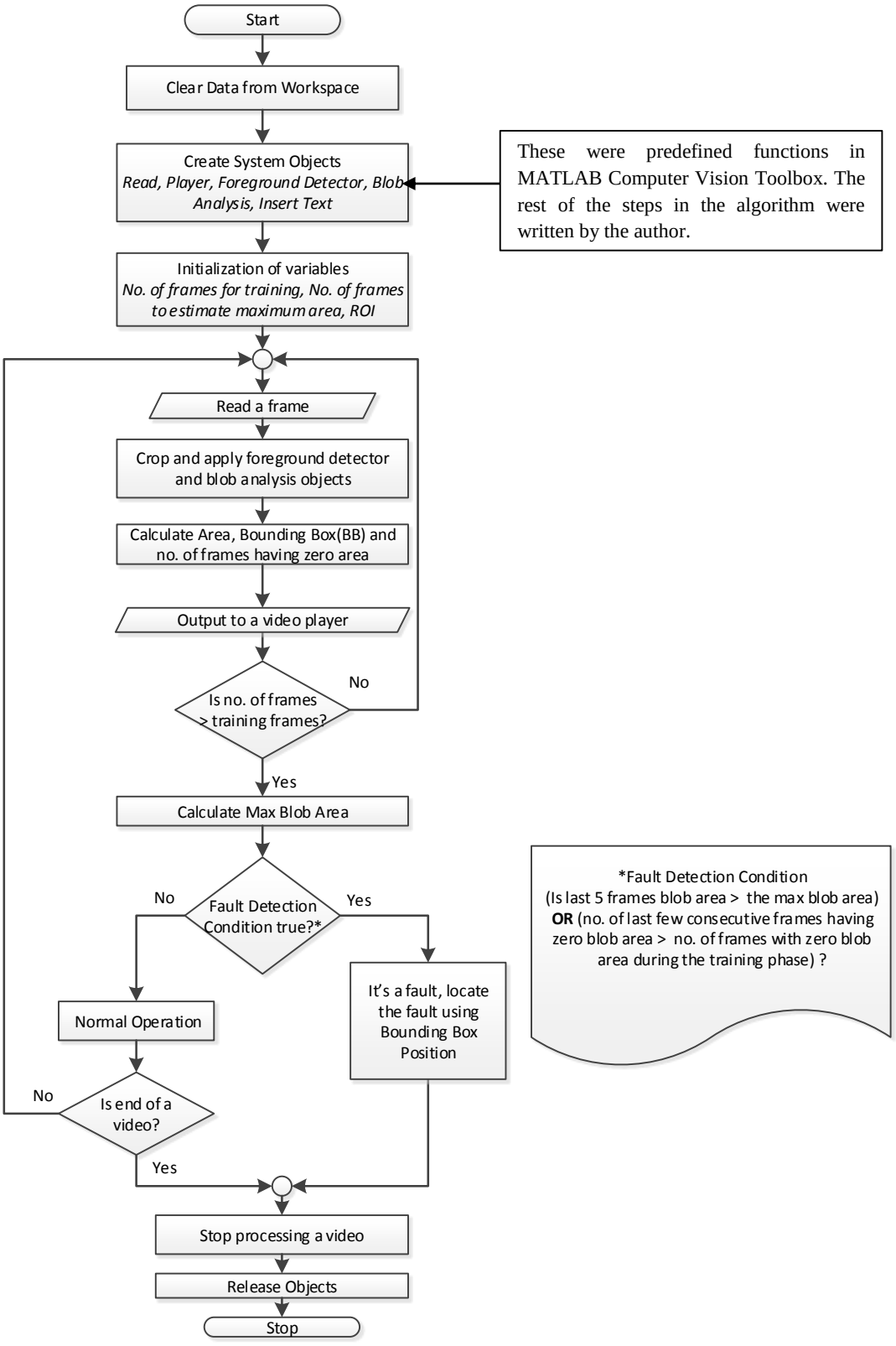


Figure 4-16 Flowchart for implementation of Method 1

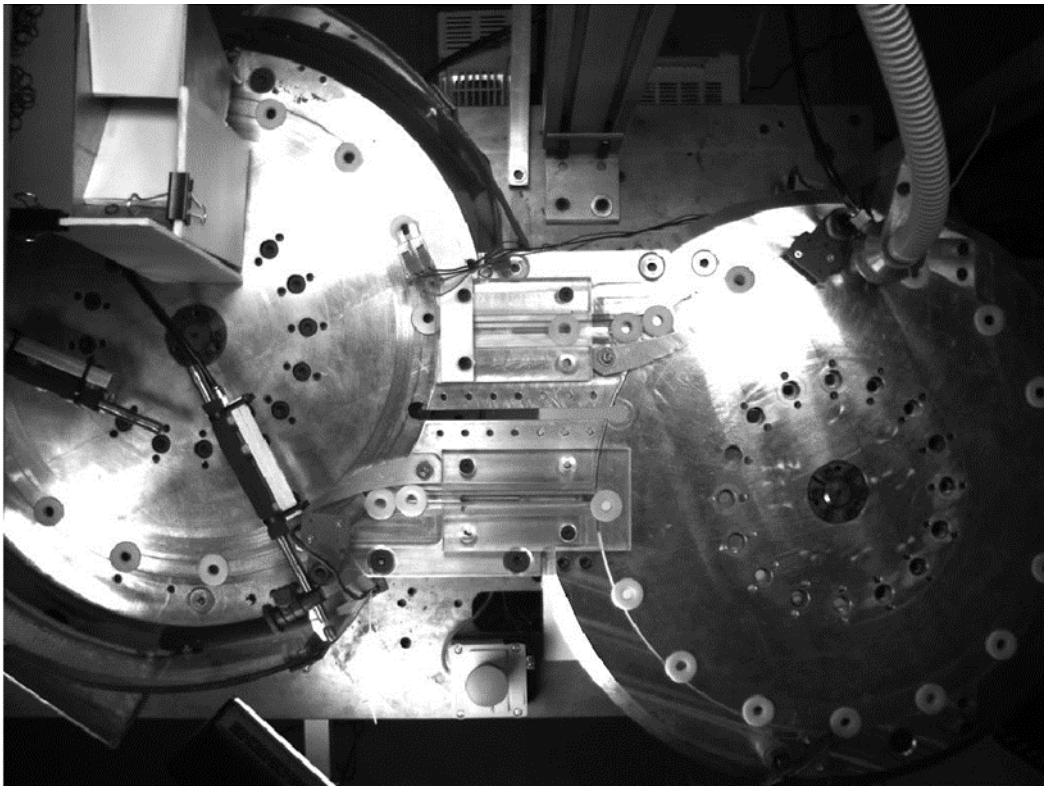


Figure 4-17 A Sample frame for transfer track region normal operation

A foreground (moving objects) is estimated from a video using Equations (4.9) to (4.16). The pixels in the detected foreground are grouped into regions with 8-connectivity using the connected component analysis. The area (in pixels) and bounding box sizes are calculated after the blob analysis of the foreground frame. Figure 4-19 represents the regions of the estimated foreground for a frame.

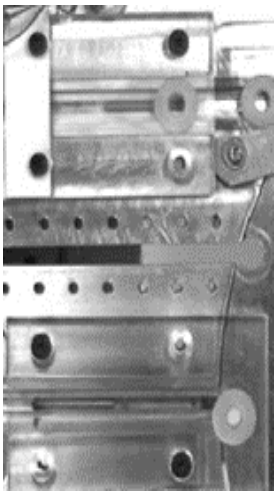


Figure 4-18 Cropped transfer track ROI to illustrate Method 1

Figure 4-19 Estimated foreground from a binary frame

The blob analysis outputs the area vector and the bounding box matrix for all the blobs in the frame. The sum of blob area for all blobs in the ROI gives the reference value of the blob area feature. This is the area (in pixels) of moving objects in a frame. For a normal operation, blob area varies but within limits between zero and the threshold value. The threshold (maximum blob area) is estimated from the first 200 frames of a video file. The 200 frames covers one full rotation of the wheels. Figure 4-20 shows the estimated threshold from the training frames.

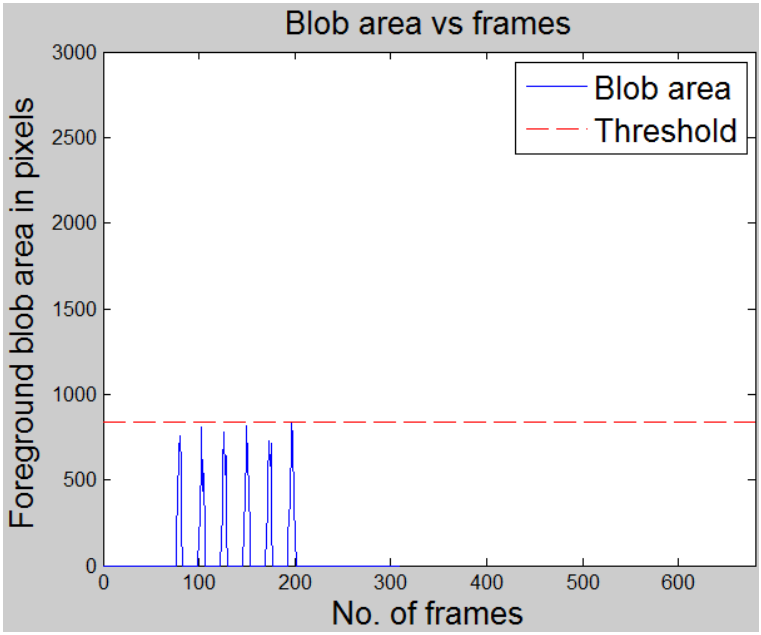


Figure 4-20 Threshold estimation from the initial training frames

In the second step, the current frame blob area is calculated and compared with the threshold. If it stays within the limits of the threshold, the operation is classified as a normal. The steps are repeated until the end of the video file or a fault, whichever occurs first. Figure 4-21 gives the plot of blob area vs number of frames for the normal operation through the transfer track region. It shows that for normal operation the blob area of the foreground stays within the limits. If it exceeds the threshold for one or two frames, and then returns within the limits, then it is mainly because of the noise or uneven motion of the objects. If the blob area for the current frame exceeds the threshold, then the method keeps track of the area for next five consecutive frames. If the blob area from subsequent frames keeps on increasing, then the condition is detected as a transfer jam. The limit of 5 consecutive frames is selected for fault confirmation because more than 5 frames added delay in fault detection, while less than 5 frames misclassified normal operation as a jam (due to a temporary jam as a result of air pressure fluctuations). The jam is then classified as either transfer 1 or transfer 2 jam by locating the bonding box coordinates of the blob area. A GUI was built in MATLAB for the method as shown in Figure 4-22. The machine operating conditions are indicated by five indicators, one for a normal operation and four for the faults.

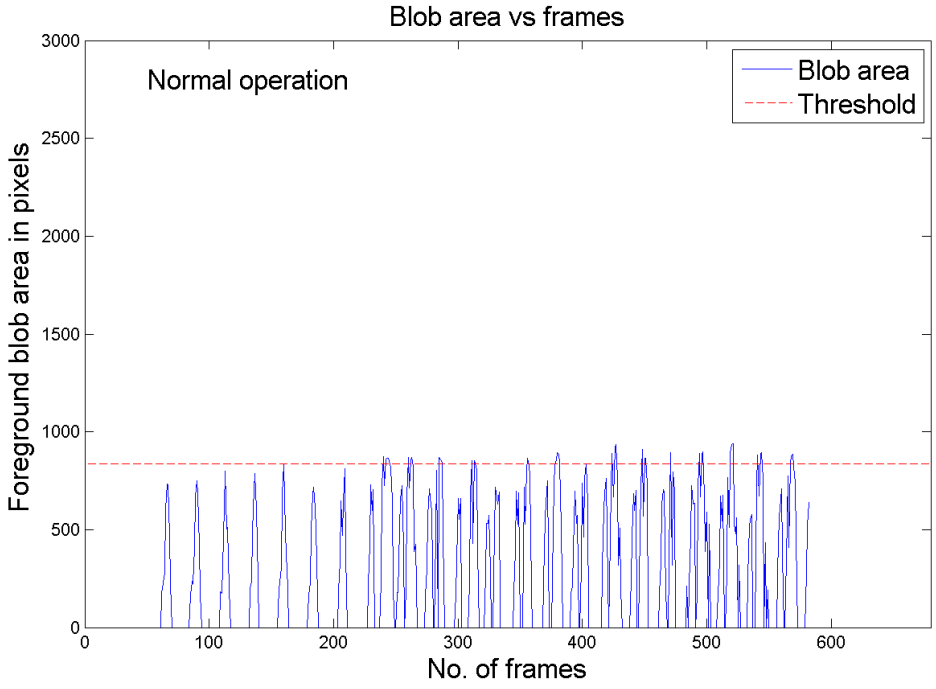


Figure 4-21 Foreground blob area plot for a normal operation

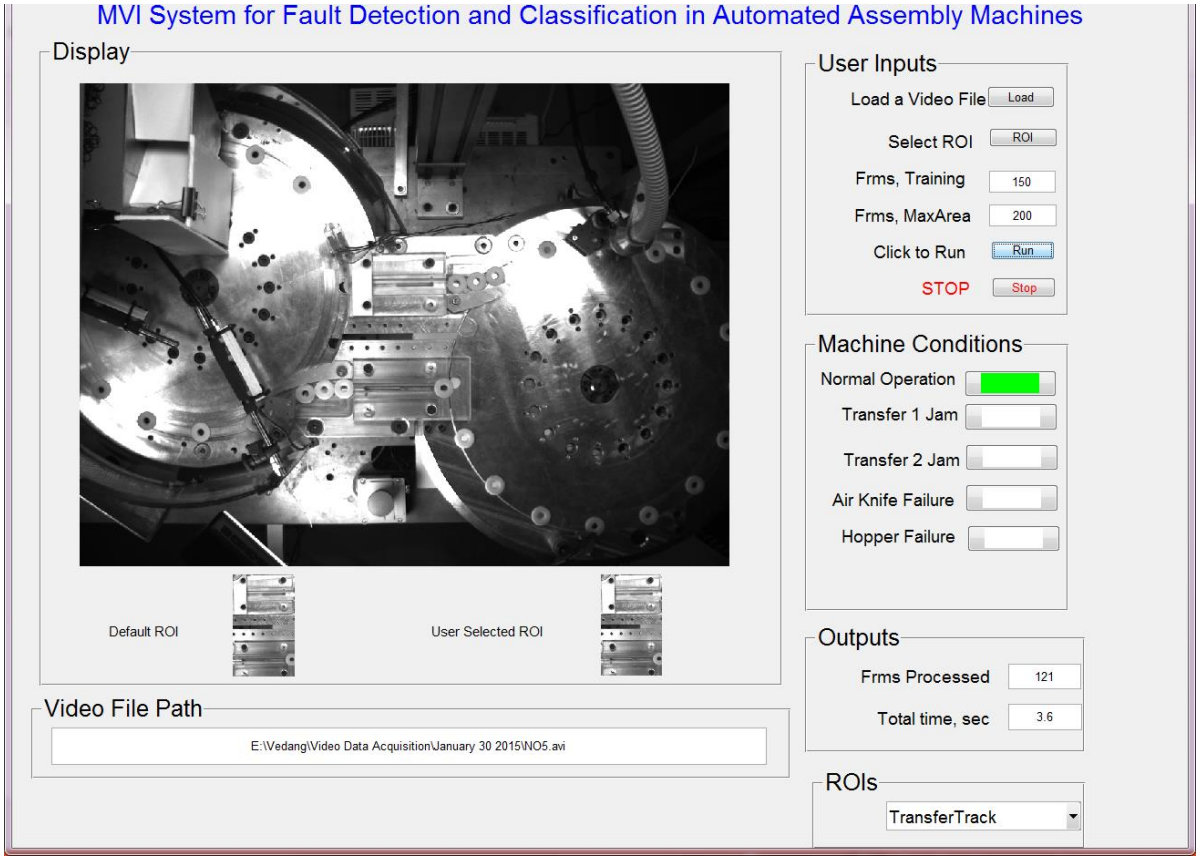


Figure 4-22 Fault detection GUI for Method 1 designed in MATLAB

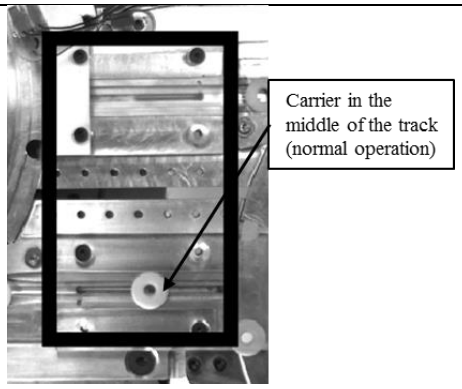
Transfer Track 1 Jam Detection with Method 1

Table 4-4 shows important events and fault signature for transfer track 1 jam detection with Method 1.

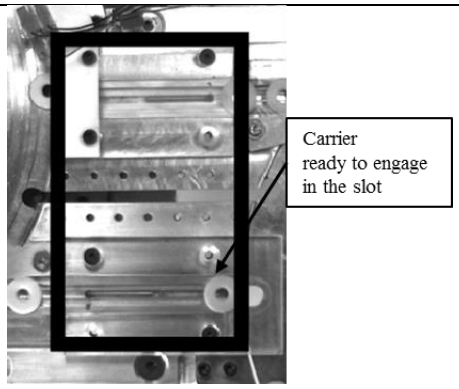
- Figure A: A carrier reached at the middle of the track in normal operation sequence.
- Figure B: The carrier was ready to engage in the slot of the secondary wheel. The fault was introduced using the HMI-PLC.
- Figure C: The carrier did not engage due to the absence of air pressure as a result of fault introduction. The blob area dropped due to lack of the motion of the carrier since Figure B.
- Figure D: Transfer jam fault detected by Method 1 and a high value (more than the threshold) of blob area was obtained due to more number of carriers in the track.
- Figure E: Continue running the machine past the fault resulted in the transfer track full of the carriers. The maximum value of blob area was obtained due to the presence of more carriers in the track.

- The above events are marked and shown in the fault signature plot. In practice, the method stopped processing the video file as soon it detected the fault. However, to get an idea of how bad a fault could have gone, the blob area till the end of video was plotted.

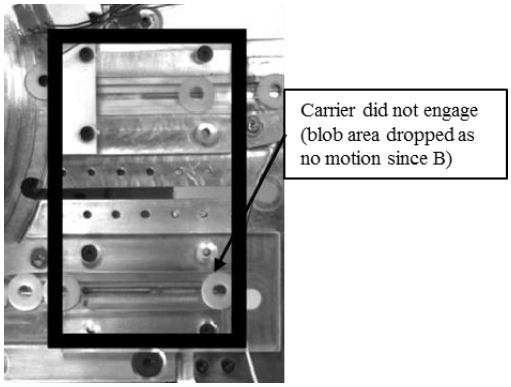
Table 4-4 Fault events and signature for transfer track 1 jam detection with Method 1



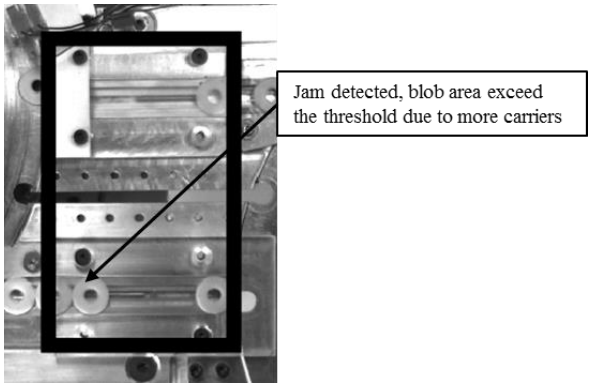
A: Normal frame (200th frame)



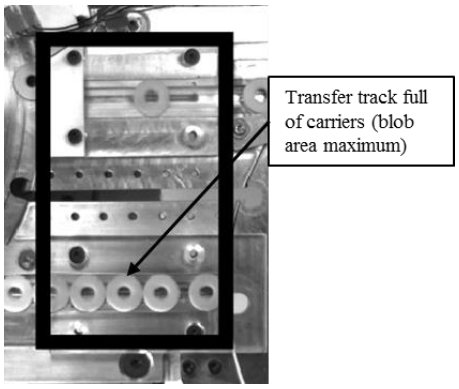
B: Fault introduction frame (300th frame)



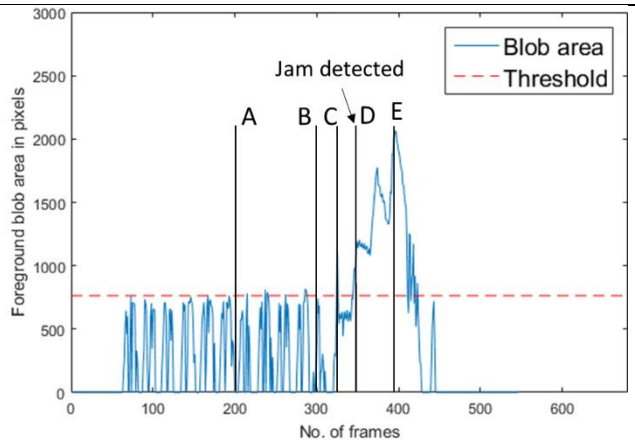
C: Fault visible frame (325th frame)



D: Fault detected frame (348th frame)



E: Fault's growth if undetected (390th frame)



Blob area plot for transfer track 1 jam

4.4 Method 2: Fault Detection with Optical Flow Estimation

Optical flow is a basic motion detection technique in video analysis. Method 2 uses optical flow for motion detection and fault classification from a video. It calculates the optical flow for a normal operation sequence, which is a measure of the movements of foreground objects. The Optical Flow Density (OFD) is computed from the optical flow of the video. The OFD, the sum of the absolute values of optical flow vectors, is a single number that represents the motion in a frame. During a normal operation sequence, the OFD stays within limits (between zero and the threshold). The OFD of the normal operation sequence is considered as the threshold (reference value). For each frame, the OFD is determined and compared with the threshold. If the current frame OFD exceeds the threshold and keeps on increasing with respect to time, then that is the indication of more number of objects in the frame. The method then keeps track of the OFD for next few frames and if the trend continues, the machine condition is considered as a fault. Larger images need more time to detect motion with the optical flow technique. Section 4.4.1 explains optical flow theory and gives the procedure to calculate the optical flow. Section 4.4.2 covers the algorithm development for Method 2.

4.4.1 Optical Flow Theory

Optical flow is a popular method for video analysis for motion detection. It finds applications in many areas of motion analysis from a video such as: motion-based segmentation, stereo from motion, video compression, traffic flow monitoring and people and vehicle tracking. It requires at least two video frames acquired at different times. It then measures how many pixels have moved from one frame to another and by what distance. An example of optical flow from a traffic video is shown in Figure 4-23. Figure 4-23(a) is the frame at time t, while Figure 4-23(b) is the frame at time $t+dt$. The optical flow computes how many pixels have moved from the frame at time t to the frame at time $t+dt$ and by what distance. The flow lines are then plotted for each pixel which has moved between the two frames. The optical flow lines are shown in white color in the figure on right. It can be seen that the flow lines are drawn only for the cars (moving objects) while no flow has been detected for the road (background). One also sees that more optical flow vector lines are visible on the car in the foreground compared to the car in the background. The amount of flow lines indicates the OFD. A high value of the OFD indicates that pixels have moved from the previous frame to the current frame with a certain area in the frame. A high degree of pixel movement is either due to the motion of a single large object that covers the whole of the region or it can be due to a number of small closely spaced objects. For the given example, it is the case of the movement of a single large object (a car). The optical flow lines can be down-sampled (i.e. selecting every alternate lines) in the case of a large number of lines for easy visualization. Hence, it can be seen that the OFD is a direct indicator of the degree of motion in a video.

Figure 4-23 Optical flow estimation from two frames: (a) frame at time t (b) frame at time t+dt (image source: MATLAB, 2014b)

Optical flow performs moving object detection from scenes by considering temporal variations in the intensity values. It is assumed that image differencing over successive pairs of frames would result in motion segmentation. However, practical motion detection is not that simple because regions of constant intensity give no sign of the motion, while object edges that are parallel to the direction of movement appear motionless as shown in Figure 4-24. Only those edges that are perpendicular to the direction of motion carry some information about the motion. It's hard to get the information about the full velocity vector due to a small aperture. This is called the aperture problem (Davies, 2005).

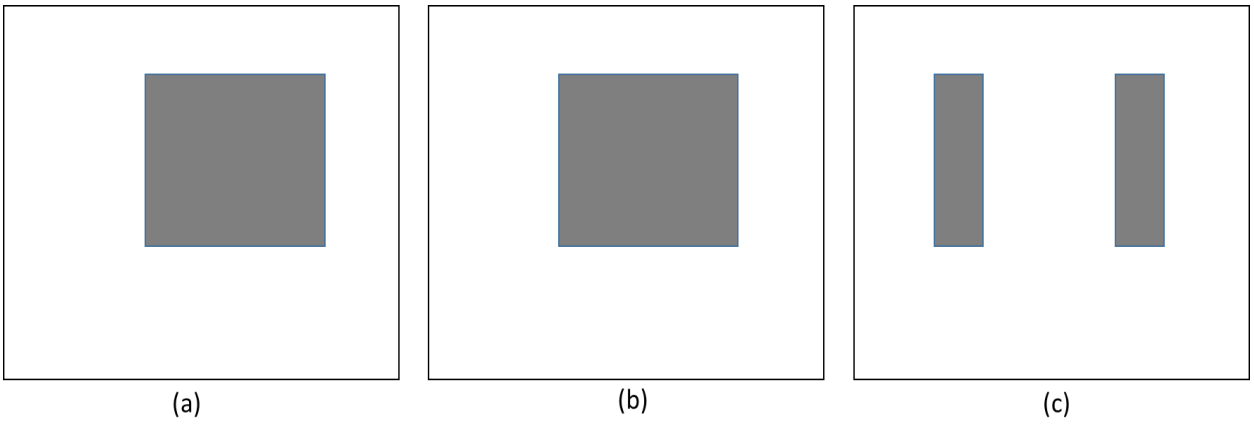


Figure 4-24 Motion estimation from frames: (a) frame at time t (b) frame at time t+dt (c) difference frame

Optical Flow Calculation

Optical flow was introduced by Horn and Schunck (1981) for computing motion between two video frames. Pixel intensity in a single frame is a 2D function of pixel coordinates (*x,y*). In the case of a video, the pixel intensity is a 3D function with coordinates (*x, y, t*), where *t* is the time or frame number in a video.

Consider two frames from a video acquired at different times, *t* and *t+dt*. The difference in time between two frames can be very small. Consider a pixel at the location (*x,y*) in the frame acquired at time *t* and its intensity is $I(x, y, t)$. Now, consider a pixel very close to the pixel in the previous frame but taken from the frame at time *t+dt* with the pixel coordinates are $(x + dx, y + dy)$. Since, the time difference between two frames is very small, the intensity for both pixels in two frames will be similar, as calculated from:

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{4.17}$$

This is called the brightness constancy assumption. It considers the same intensity between two frames for pixels that are very close to each other. Taylor series expansion of right side of Equation (4.17) results in:

$$I(x, y, t) = I(x, y, t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt + \text{higher order terms} \tag{4.18}$$

Simplifying Equation (4.18) and considering lower order terms only:

$$I_x dx + I_y dy + I_t dt = 0 \tag{4.19}$$

where, I_x, I_y and I_t are spatiotemporal image brightness derivatives with respect to x, y and t respectively. Dividing each term of Equation (4.19) by dt,

$$I_x u + I_y v + I_t = 0 \tag{4.20}$$

where, $u = \frac{dx}{dt}$, optical flow in x-direction and v= $\frac{dy}{dt}$, optical flow in y-direction. This equation is called the optical flow constraints equation. Optical flow for pixels between two frames in shown in Figure 4-25.

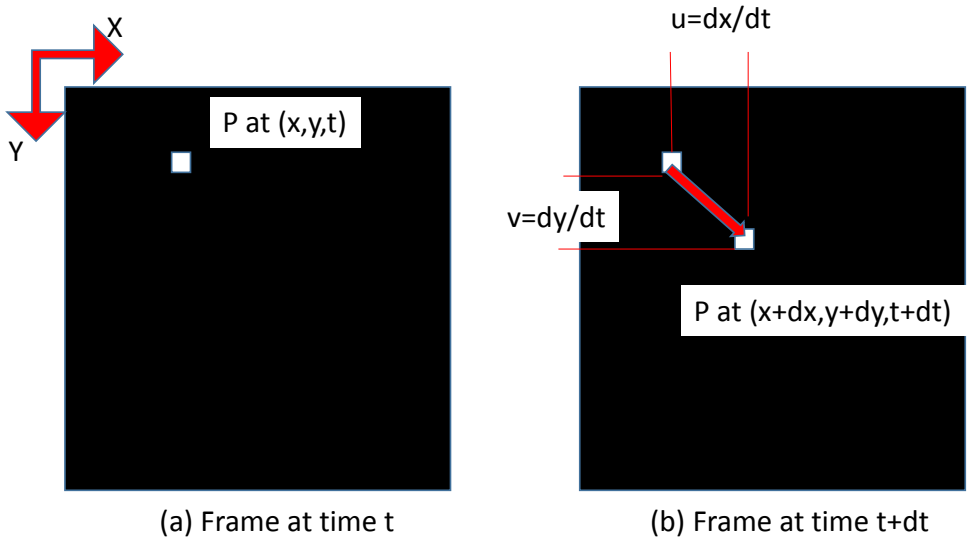


Figure 4-25 Optical flow in *x* and *y* direction

Equation (4.20) is under constrained as there is one equation and there are two unknowns (*u*, *v*). The equation can be solved in several ways. Two popular approaches are:

- Horn- Schunck method (Horn & Schunck, 1981)
- Lukas-Kanade method (Lucas & Kanade, 1981)

The Horn-Schunck method is considered for the fault detection problem in this thesis. This method solves the equation using an optimization approach by assuming the smooth optical flow over the entire image. It computes the optical flow by minimizing the global energy equation, given as:

$$E = \iint (I_x u + I_y v + I_t)^2 dxdy + \alpha \iint \left\{ \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \right\} dxdy \quad (4.21)$$

Equation (4.21) has two terms, the first term is based on the brightness constancy and the second term is based on the smoothness constraint and α is the scaling factor for the second term.

The Horn-Schunck method minimizes a global energy equation and solves it using the calculus of variations. The global energy is minimized by equating derivatives of Equation (4.21) with respect to *u* and *v* to zero as per the following two equations:

$$\left(I_x u + I_y v + I_t\right) I_x + \alpha\,(\nabla^2 u) = 0 \tag{4.22}$$

$$\left(I_x u + I_y v + I_t\right) I_y + \alpha\,(\nabla^2 v) = 0 \tag{4.23}$$

where, ∇^2 is the Laplacian operator and $\nabla^2 u$ and $\nabla^2 v$ are the Laplacians of *u* and *v*, respectively, as given by:

$$\nabla^2 u = u_{xx} + u_{yy} \tag{4.24}$$

$$\nabla^2 v = v_{xx} + v_{yy} \tag{4.25}$$

where, u_{xx} is second derivative of u with respect to x and u_{yy} is the second derivate of u with respect to *y*. The discrete version of Equations (4.22) and (4.23) can be given as:

$$\left(I_x u + I_y v + I_t\right) I_x + \alpha\,(u - u_{av}) = 0 \tag{4.26}$$

$$\left(I_x u + I_y v + I_t\right) I_y + \alpha\,(v - v_{av}) = 0 \tag{4.27}$$

where α is a scaling factor, u_{av} and v_{av} are average of neighbors of (*u*,*v*). These are now two equations and two unknowns. These equations can be solved using the calculus of variations to obtain values for *u* and *v* as given by the following equations:

$$u = u_{av} - I_x \frac{A}{B} \tag{4.28}$$

$$v = v_{av} - I_y \frac{A}{B} \tag{4.29}$$

where, $A = I_x u_{av} + I_y v_{av} + I_t$ and $B = \alpha + I_x^2 + I_y^2$

The above equations are solved iteratively and *u* and *v* are obtained as per the following equations:

$$u_{x,y}^{k+1} = u_{x,y}^{-k} - \frac{I_x[I_x\, u_{x,y}^{-k} + I_y v_{x,y}^{-k} + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{4.30}$$

$$v_{x,y}^{k+1} = v_{x,y}^{-k} - \frac{I_y[I_x\, u_{x,y}^{-k} + I_y v_{x,y}^{-k} + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{4.31}$$

In the above equations, $[u_{x,y}^k, v_{x,y}^k]$ is the velocity estimate for the pixel at location (x, y) and $[u_{x,y}^{-k}, v_{x,y}^{-k}]$ is the neighborhood average of $[u_{x,y}^k, v_{x,y}^k]$. The initial velocity is assumed as zero.

The five steps to obtain optical flow by Horn-Schunck method are given below (MATLAB, 2014b):

1. Compute image derivatives I_x and I_y using the Sobel operator (as given below) and its transposed form for each pixel in the first image.

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

2. Compute the time derivative I_t between two images using the kernel, $[-1 \quad 1]$.
3. Assuming initial velocity to be zero, compute the average velocity for each pixel using the convolution kernel,

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

4. Iteratively solve for *u* and *v*.

The optical flow for each frame of a video is calculated using the above steps by implementing a code in MATLAB. The optical flow is output as horizontal and vertical components in complex forms. The optical flow density (OFD), is the absolute value of optical flow as determined for each frame. The OFD is used as a feature to detect and classify faults from videos with method 2.

4.4.2 Implementation of Method 2

The application of Method 2 to the problem at hand was not straight forward. This section describes the technique that was developed to enable implementation. The implementation for Method 2 is given as steps in Table 4-5 and as a flowchart in Figure 4-26. The MATLAB code information is provided in Appendix C. The steps assume normal operation through the transfer track region.

Table 4-5 Steps for implementation of Method 2

Stage	Step No.	Description
Fault Detection	1	Clear workspace variables
	2	Create system objects *(file read, player, optical flow, shape inserter)*
	3	Initialize variables *(No. of frames for training, optical flow matrix , ROI)*
	4	Read a frame at a time, after the last frame read go to ***step 15***
	5	Crop ROI and detect optical flow with the optical flow object
	6	Subsample optical flow, compute absolute value, Optical Flow Density (OFD) and no. of frames having zero OFD
	7	Insert flow lines using the shape inserter and output a frame to the player
	8	Is no. of frames greater than no. of frames for training? If No, go to ***step 4***. If yes, continue to next step
	9	Calculate the OFD for a normal operation (2 × average value of OFD)
	10	(Is last 5 frames OFD greater than the normal OFD) OR (no. of last few consecutive frames having zero OFD is greater than the no. of frames with zero OFD during the training phase)? If no, go to ***step 11***. If yes, go to ***step 12***
Fault Classification	11	Set normal operation LED on and go to ***step 4***
	12	Is a flow density in the lower part of the ROI greater than the upper part? If yes, go to ***step 13*** else go to ***step 14***
	13	Set transfer 1 jam LED on, stop processing and go to ***step 15***
	14	Set transfer 2 jam LED on, stop processing and go to ***step 15***
	15	Release all objects from the memory

The first step is to create a system object to read a video file. The video is read frame by frame and each frame is cropped to the ROI defined by the user. For the transfer track region, the cropped ROI is shown in Figure 4-27. For each frame the optical flow is calculated with the Horn-Schunck method by assuming brightness constancy and smoothness constraint. That means all pixels on a moving object have the same optical flow. The reference frame delay is set to 1, hence the optical flow is calculated between every consecutive frames. The maximum number of iterations to perform in the iterative computation is set to 10. The optical flow is output in the form of horizontal and vertical components in the complex form.

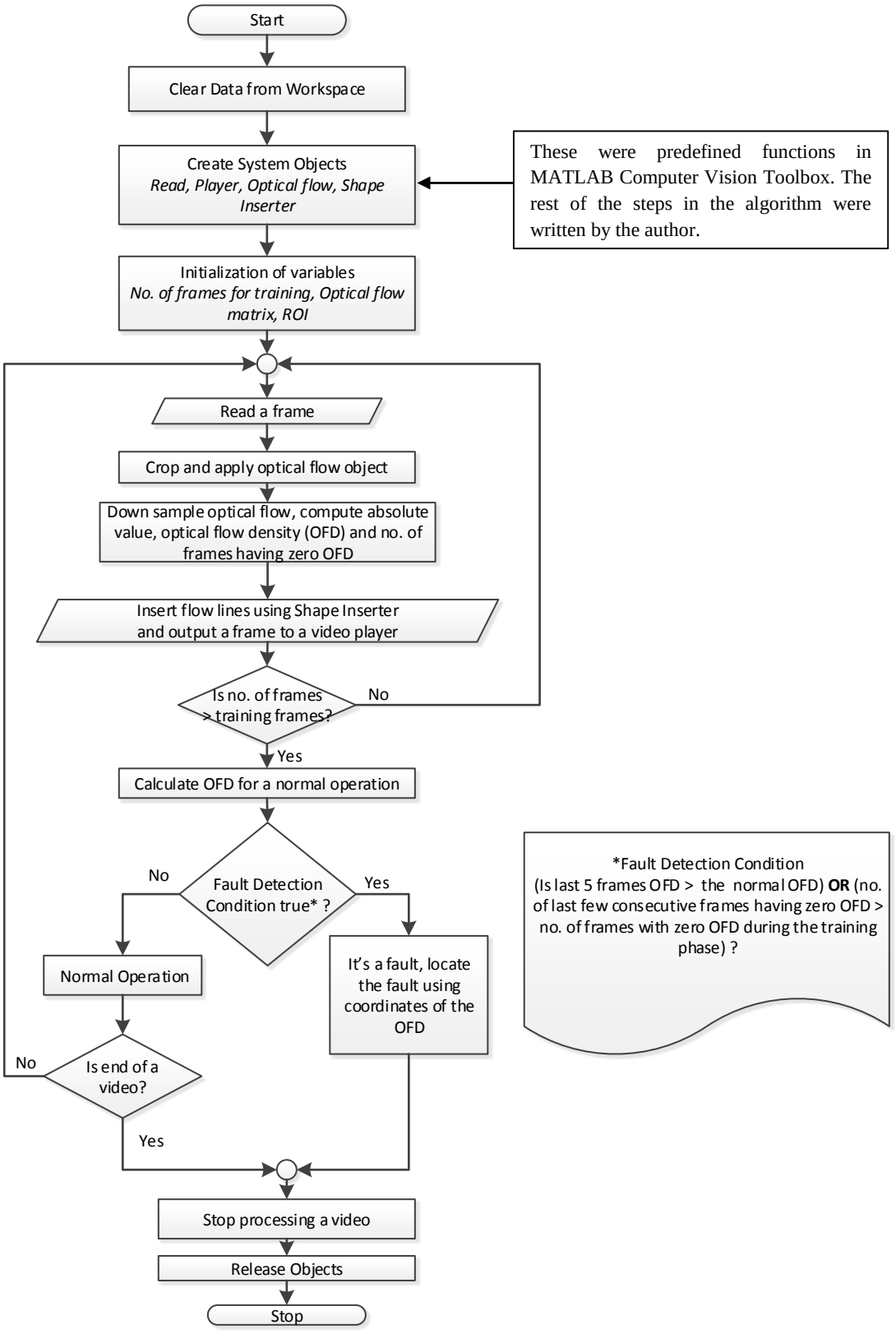


Figure 4-26 Flowchart for implementation of Method 2

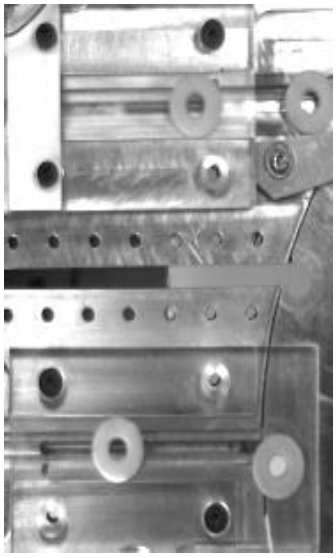


Figure 4-27 Cropped transfer track ROI to illustrate Method 2

Optical flow is sensitive to very small motions such as vibrations in the machine. Hence, large number of optical flow vectors are obtained for a given frame. Optical flow vectors are subsampled (to reduce the flow vectors due to noise) by considering every 3rd vector along image width and height. Additionally, the subsampled optical flow vectors are very small numbers in a complex form. Therefore, they are scaled by a factor of 10. The resulting flow lines are plotted in a frame using lines joining each pixel to its position in the previous frame. Figure 4-28 shows optical flow between two frames (frame no. 150 and frame no. 151). The flow lines are visible in white color in Figure 4-28 (b). The motion from a previous frame to the current frame is plotted in the current frame using white lines. Due high speed of a carrier through the pneumatic transfer track, the traces of motion can be seen in the transfer track 1 region.

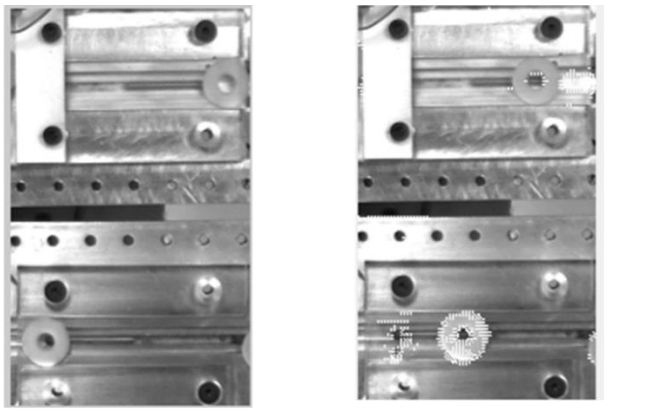


Figure 4-28 Optical flow in a normal operation through the transfer track ROI

Optical flow values are represented in a complex form ($u + vi$); therefore, the absolute value is calculated for each complex vector. The OFD is the sum of these absolute values for a given frame. The OFD is directly proportional to the number of moving objects in a frame. The reference value (threshold) of OFD is estimated from the first 200 training frames (one full rotation of the wheels) of a video file. For a normal operation, OFD stays within the limits (0 and threshold). Figure 4-29 shows the estimated threshold (2 × average value of the OFD) from the training frames.

In the second step, OFD of the current frame is computed and compared with the threshold. If it stays within limits between zero and the threshold, the operation is classified as a normal. The steps are repeated until the end of the video or a fault, whichever occurs first. Figure 4-30 depicts a full plot of OFD vs number of frames for the normal operation through the transfer track region. It shows that for normal operation, the OFD stays within limits. It can also be noted that the OFD signal is very noisy as it is sensitive to every small movement in the frame. Even though the OFD exceeds the limit in some frames, the condition is not considered as a fault because the OFD does not stay high and it returns within the limits in the next frame.

If the OFD of the current frame exceeds the threshold, then the method keeps track of the OFD for the next 5 consecutive frames. If the OFD of the subsequent frames keeps on increasing, then the condition is detected as a transfer jam. The limit of 5 consecutive frames is selected for fault confirmation because more than 5 frames added delay in fault detection, while less than 5 frames misclassified normal operation as a jam. A GUI, similar to Method 1, was written in MATLAB for the Method 2. The machine operating conditions are indicated by five indicators, one for a normal operation and four for the faults.

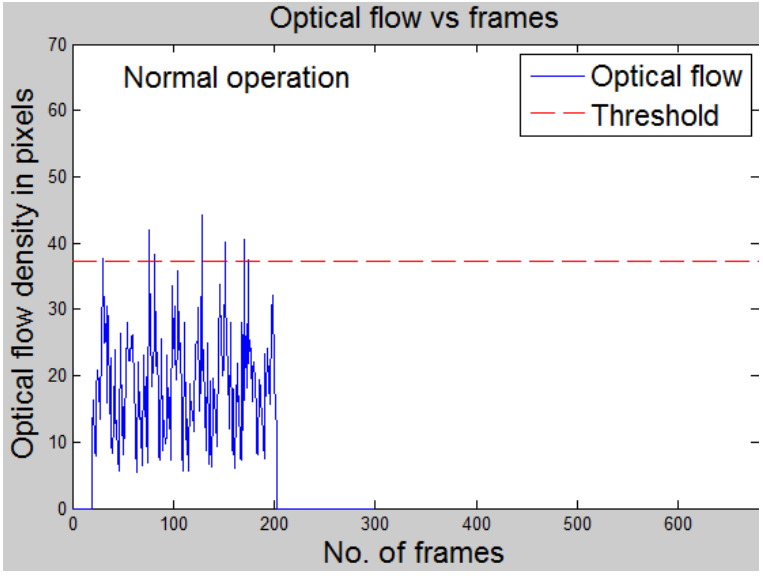


Figure 4-29 OFD threshold estimation from the initial training frames

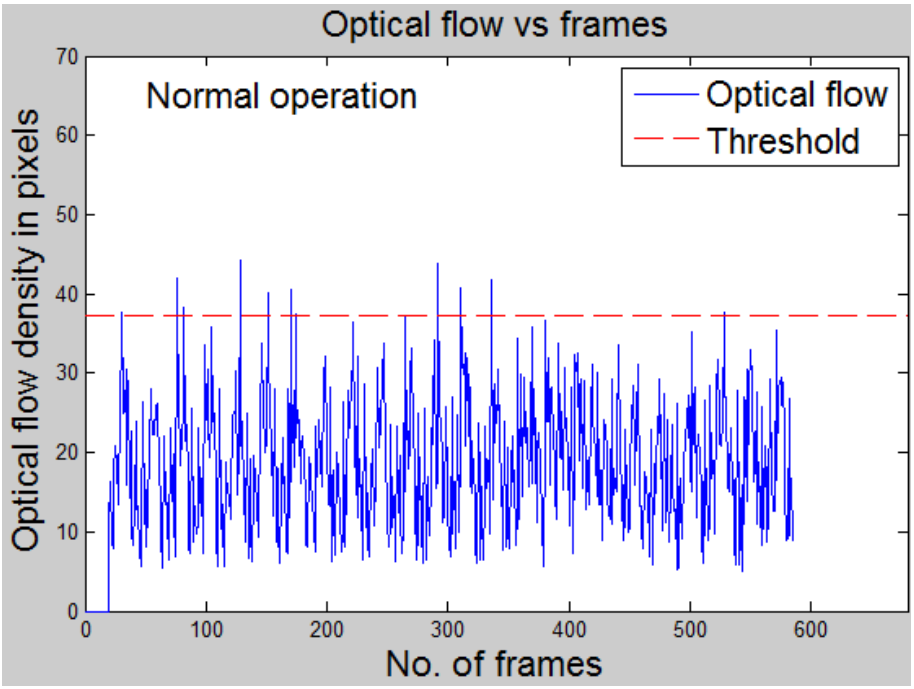


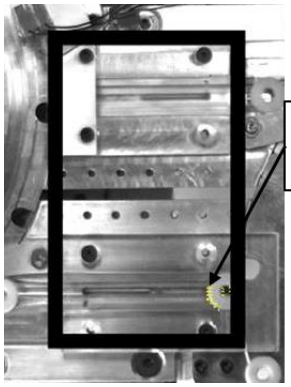
Figure 4-30 OFD for a normal operation through the transfer track ROI

Transfer Track 1 Jam Detection with Method 2

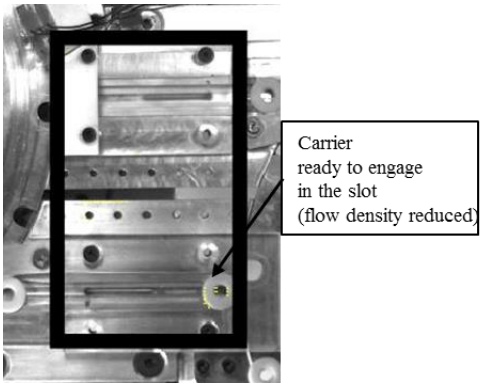
Table 4-6 shows important events and fault signature for transfer track 1 jam detection with Method 2.

- Figure A: A carrier reached at the end of the track in normal operation sequence.
- Figure B: The carrier was ready to engage in the slot of the secondary wheel. The fault was introduced using the HMI-PLC.
- Figure C: The carrier did not engage due to the absence of air pressure as a result of fault introduction. The OFD reduced due to lack of the motion of the carrier since Figure B.
- Figure D: Transfer jam fault detected by Method 2 and high density of the OFD was obtained due to more number of carriers in the track.
- Figure E: Continue running the machine past the fault resulted in the transfer track full of the carriers. The OFD reduced due to less movement of the carriers in the track.
- The above events are marked and shown in the fault signature plot. In practice, the method stopped processing the video file as soon it detected the fault. However, to get an idea of how bad a fault could have gone, the OFD till the end of video was plotted.

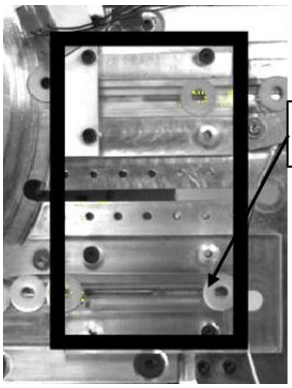
Table 4-6 Fault events and signature for transfer track 1 jam detection with Method 2



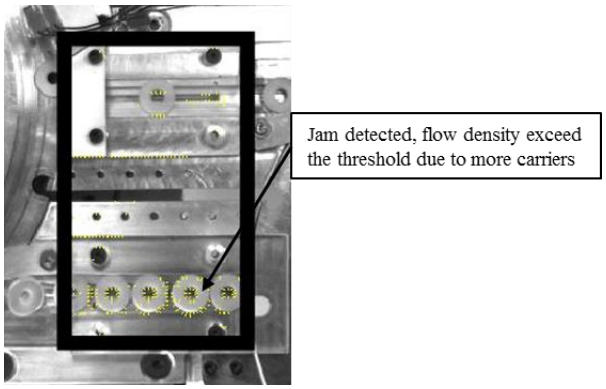
A: Normal frame (200th frame)



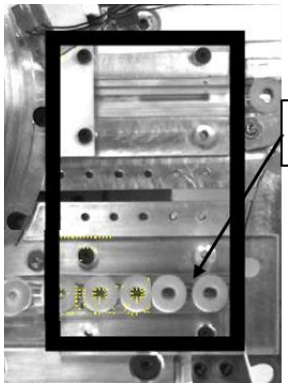
B: Fault introduction frame (300th frame)



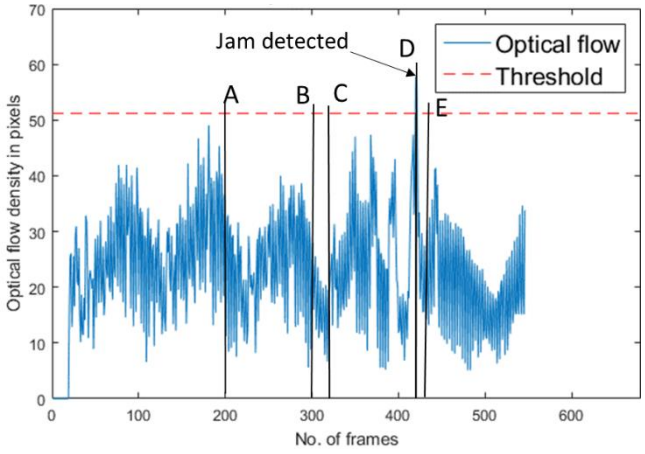
C: Fault visible frame (325th frame)



D: Fault detected frame (420th frame)



E: Fault's growth if undetected (435th frame)



OFD plot for transfer track 1 jam

4.5 Method 3: Fault Detection with Foreground Running Average Area

In this method, a fault in the machine is detected by measuring an area of a processed foreground from a ROI. The measured area is then compared against a threshold and if it exceeds the threshold for a few consecutive frames, the machine condition is classified as a fault.

The first step in the method is an estimation of a background frame. The machine has a rotary motion, hence both the primary and the secondary wheels along with fasteners and assemblies move in the scene. Capturing an image without the assemblies on the wheels for background estimation did not work for this method. It gave a steady constant background frame which was a poor estimate of the background when the machine was in motion. Hence, there was a need for a technique that could properly estimate a background from a video and is robust to the motion and minor changes in the lighting conditions. A good background image must be updated regularly to adapt to the varying illumination conditions or camera oscillations. Methods such as a Mixture of Gaussians or Mean-shift are popular, but they require high computational efforts and processing time (Stauffer & Grimson, 1999). Usamentiaga et al. (2013) used running average background estimation for the application of jam detection on steel pickling lines. Their method worked for high frame rates with a limited memory requirement. They used the adaptive background estimation method that was implemented using the following equation:

$$B_t = \alpha I_t + (1 - \alpha) B_{t-1} \tag{4.32}$$

where B_t is a background frame and α is an empirical weight used as a tradeoff between stability and quick updating.

Due to the large number of rotating components on the machine and high speed of movement of the assemblies through the transfer tracks, the above formula did not work well for the O-ring machine application. Therefore, running average background estimation using a different approach (as explained below) is considered. Description of the main steps used in the Method 3 is given in the next section.

4.5.1 Background Estimation and Foreground Running Average Area Feature Extraction

A fast and efficient technique for background estimation is implemented for Method 3. The method uses preprocessing steps, prior to background estimation, such as spatial Gaussian filter for noise removal and histogram equalization. It reads each image frame and implements the above mentioned preprocessing steps. The machine was operated under a low lighting condition (overhead lights off) to minimize the effect of ambient light. Moreover, to avoid specular reflections from the shiny wheel surfaces, dark field lighting was preferred for video data acquisition. As a result of low light illumination, the video was sensitive to noise. Hence, the first step is the filtering of each frame with a spatial Gaussian filter to minimize the effect of noise.

Image filtering is a mathematical operation between an image and a sub image, where the sub image is called a filter or mask. The filtering operation performed directly on pixel intensity values is called spatial filtering. The sub image is of a smaller size than the main image. The size of the sub image is the same as the size of each pixel's neighborhood considered for processing. The mechanics of image filtering is explained given by:

$$J(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) \times I(x+s, y+t) \tag{4.33}$$

where, I is the input image of size $M \times N$, w is the filter image of size $m \times n$, and J is the filtered image, $a = \frac{m-1}{2}$ and $b = \frac{n-1}{2}$. At each pixel in the input image, its neighborhood is considered and processed by multiplying with the corresponding filter image coefficients. The filter image is then moved to a new pixel location and the response is calculated. To generate the complete filtered image, Equation (4.33) must be applied for $x = 0, 1, 2, \ldots\ldots M-1$ and $y = 0, 1, 2, \ldots\ldots N-1$. For fault detection with Method 3, each frame in a video is preprocessed by Gaussian low pass spatial filter as implemented using Equation (4.33).

The second preprocessing step is histogram equalization. A histogram of a digital image is a plot of the number of pixels having a particular gray level vs number of gray levels in the image. Dividing each pixel's number of occurrences by the total number of pixels, a normalized histogram is obtained. It gives the probability of occurrence for each gray level in the image. There is a direct correlation between image appearance and its histogram. For example, a dark image has its histogram concentrated on left side of the gray level values. Similarly, an image with high contrast and brightness, has the histogram that occupies the entire gray scale range and it has an equal distribution. Therefore, modifying image histogram, changes its appearance. For this thesis, as mentioned previously, the frames are acquired in a low lighting condition. Hence, the frames have histograms that do not cover the entire gray scale span and do not have equal gray level distributions.

Histogram equalization is the gray level transformation function that uses gray level probability values and tries to redistribute gray level values such that the resulting image has a histogram that spans the full range and it has nearly equal (not always) probability of occurrences for each gray level. The histogram equalization function is derived from cumulative probability distributions of gray values in the image. For normalized image histogram, its equalization function can be a function given by:

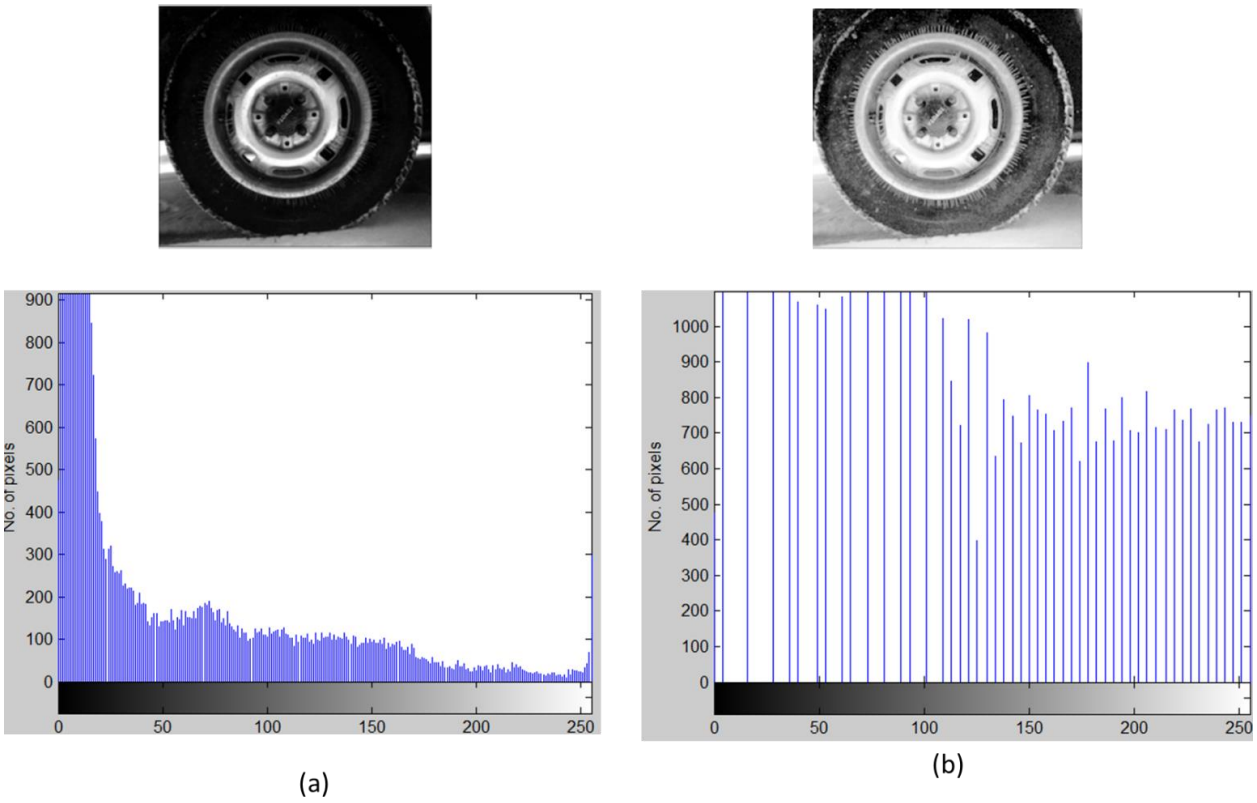


Figure 4-31 Histogram equalization example: (a) original image (b) hisogram equalized image

(image source MATLAB, 2014b)

$$s = T(r) \qquad 0 \le r \le 1 \tag{4.34}$$

where, r is input image gray value, s is out image gray value and T is the transformation function. The transformation function should satisfy the following two conditions for histogram equalization:

(1) $T(r)$ is single-valued and monotonically increasing in the interval, $0 \le r \le 1$.

(2) $0 \le T(r) \le 1$ for $0 \le r \le 1$

The first condition is needed to guarantee that the inverse transform will exist and the monotonicity condition preserves the increasing order from black to white in the output image. The second condition guarantees that the output gray levels will be in the same range as the input gray levels. Figure 4-31 shows an example result of histogram equalization.

Figure 4-31(a) shows the original image and its histogram. Figure 4-31(b) shows the image after histogram equalization and its histogram. It can be seen how histogram equalization redistributes gray levels equally

in the image. Minor details can be seen in the second image, which are not visible in the first image due to its dark appearance.

The third step of the method is background estimation by image averaging and calculating the running average. In this step, a background frame is obtained by averaging the last 20 frames. The running average method with a filter size of 20 is used to estimate the background from the video. A filter size less than 20 resulted in improper estimate of the background and higher than 20 caused delays in processing. The background at any time t was estimated using Equation (4.35). The background estimation is dynamic, in the sense that the estimated background is updated after every 20 frames.

$$B_t = \left(\sum_{i=1}^{20} I_i\right)/20 \tag{4.35}$$

The method is very fast and has low memory requirements compared to advanced background estimation methods.

The fourth step is foreground, F, detection by background subtraction using:

$$F_t = |I_t - B_t| \tag{4.36}$$

where, I_t is the frame at time *t*, B_t is the estimated background and F_t is the resulting foreground.

The foreground contains the moving objects, such as carriers and assemblies and noise (unwanted details). The image is then segmented using thresholding to obtain a binary frame. As a result of thresholding, only black and white pixels are available in the image. To further reduce the noise and to get attributes of only the moving objects, morphological operations are performed on the segmented binary image prior to feature extraction.

Morphology is the study of the form and structure of objects. Mathematical morphology is a tool based on set theory that is used to extract image components that are useful in the representation and description of a region's shape, such as boundaries, skeletons, regions, and the convex hull (Gonzalez & Woods, 2002). Morphological operations are based on basic concepts from set theory. Dilation and erosion are two fundamental operations in morphological processing. To some extent morphological operations are similar to spatial filtering, in which an image is processed by another sub image. In case of morphological operations, the sub image is called a structuring element.

Dilation of image I with a structuring element s is the morphological operation denoted by, $I \oplus s$ and defined by:

$$I \oplus s = \{z \mid [(\hat{s})_z \cap I\,] \subseteq I\} \tag{4.37}$$

where z^2 is the two dimensional integer space, $\hat{s}$ is the reflection of s with respect to its reference point and $(\hat{s})_z$ is the translation of the reflection of s by z.

Equation (4.37) explains that the dilation of I by s is the set of all displacements, z, such that $\hat{s}$ and I overlap by at least one element. The dilation increases the overall image dimension. Therefore, it is preferred for bridging gaps in the image.

Erosion of an image I with a structuring element s is the morphological operation denoted by, $I \ominus s$ and defined by:

$$I \ominus s = \{z \mid (s)_z \subseteq I\} \tag{4.38}$$

where z^2 is the two dimensional integer space, $(s)_z$ is the translation of s by z.
This equation means that the erosion of I by s is the set of all points, z, such that s translated by z is contained in I . The erosion reduces the size of the objects in the image. Therefore, it is preferred for removing small irrelevant details in the image. Anything in the image with size smaller than the structuring element will be removed by this erosion operation.

Dilation and erosion are dual of each other with respect to complementation and reflection as given by:

$$(I \ominus s)^c = I^c \oplus \hat{s} \tag{4.39}$$

Morphological opening and closing operations are constructed using dilation and erosion in a specific order. Morphological opening results in smooth contours, it eliminates thin protrusions and it breaks narrow connections in an image. On the other hand, morphological closing also results in smooth image contours but it fuses narrow connections, eliminates small holes and fills gaps in the contour. Morphological opening of I by s is given by Equation (4.40) and morphological closing can be obtained by Equation (4.41):

$$I \circ s = (I \ominus s) \oplus s \tag{4.40}$$

$$I \cdot s = (I \oplus s) \ominus s \tag{4.41}$$

Similar to dilation and erosion, morphological opening and closing are also duals of each other with respect to complementation and reflection, as given by:

$$(I \cdot s)^c = I^c \circ \hat{s} \tag{4.42}$$

Morphological operations are shown in Figure 4-32 where: (a) is the original image and (b) is the structuring element. The morphological opening operation results in removal of small connections in the image and only four corner details are visible in (c). By contrast, the morphological closing operation gives (d), which has filled the gap in the square and increased the overall object area. The actual size of the structuring element is smaller than its appearance in the image.

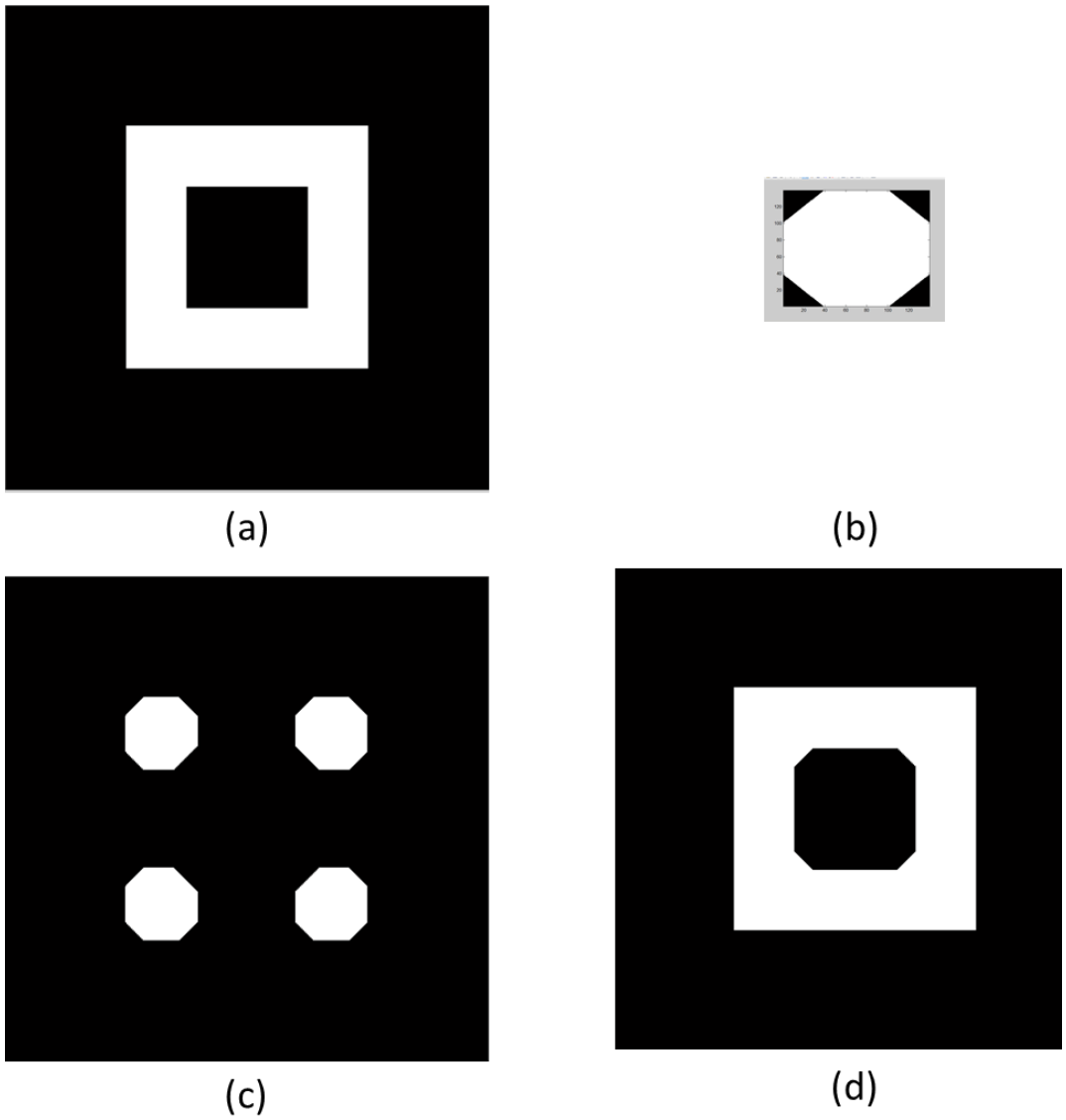


Figure 4-32 Morphological operations: (a) original image, (b) structuring element, (c) opening and (d) closing

After morphological operations by the algorithm, the clusters of white pixels are available that are corresponding to the moving objects. As the second step, an area of the moving objects is calculated in each frame by counting number of white pixels. The area is stored in a vector and the running average area is calculated from the vector using a running average filter. Running average area from the last 7 frames is computed, using Equation (4.43). The limit of 7 frames is selected for running average area because more than 7 frames added delay in fault detection while less than 7 frames misclassified normal operation as a faulty operation. The result is used as a feature to classify the machine condition into normal or faulty condition.

$$A_t = \left(\sum_{i=1}^{7} A_i\right)/7 \tag{4.43}$$

where, A_i is the foreground object area in a frame and A_t is the running average area from the last 7 frames.

Morphological opening and closing operations of the segmented foreground results in only large moving objects visible in the frame. The morphological operations remove small irrelevant details such as fasteners and other small moving objects from, the segmented foreground image. An area of moving object in the morphologically processed image is calculated by simply counting the number of white pixels. The area is then stored in a vector where one value is stored for each frame of the video file. The area may have zero values for the frames where no carriers and O-rings are visible in the frame. Hence, the foreground area plot is not smooth for the operation. To overcome this issue and to reduce the effect of noise in the calculation of the foreground area, the running average of the foreground area vector is computed. The running average vector is created where each element of the vector contains the running average area for a frame.

4.5.2 Implementation of Method 3

The application of Method 3 to the problem at hand was not straight forward. This section describes the technique that was developed to enable implementation. The implementation for Method 3 is given as steps in Table 4-5 and as a flowchart in Figure 4-33. The MATLAB code information is provided in Appendix C. The steps are explained assuming normal operation through the transfer track region.

Table 4-7 Steps for implementation of Method 3

Stage	Step No.	Description
Fault Detection	1	Clear workspace variables
	2	Create system objects *(file read and a player)*
	3	Initialize variables *(Running average filter size, structuring element, ROI)*
	4	Read a frame at a time, after the last frame read go to ***step 19***
	5	Crop ROI and apply: 5×5 Gaussian low pass filter for noise removal and histogram equalization
	6	Estimate a background frame by running average of 20 frames
	7	Calculate a foreground frame by subtracting the background frame
	8	Perform segmentation with thresholding
	9	Remove noise and stray objects by morphological closing and opening
	10	Calculate a running average area and no. of frames having zero running average area
	11	Output a frame to the player and plot areas
	12	Is no. of frames greater than no. of frames for training? If No, go to ***step 4***. If yes, continue to next step
	13	Calculate the normal operation running average area (maximum value)
	14	(Is last 7 frames running average area greater than the normal running average area) OR (no. of last few frames having zero running average area is greater than the no. of frames with zero running average area during the training phase)? If no, go to ***step 15***. If yes, go to ***step 16***
Fault Classification	15	Set normal operation LED on and go to ***step 4***
	16	Is avg. T1 area greater than T2 area? If yes, go to ***step 17*** else go to ***step 18***
	17	Set transfer 1 jam LED on, stop processing and go to ***step 19***
	18	Set transfer 2 jam LED on, stop processing and go to ***step 19***
	19	Release all objects from the memory

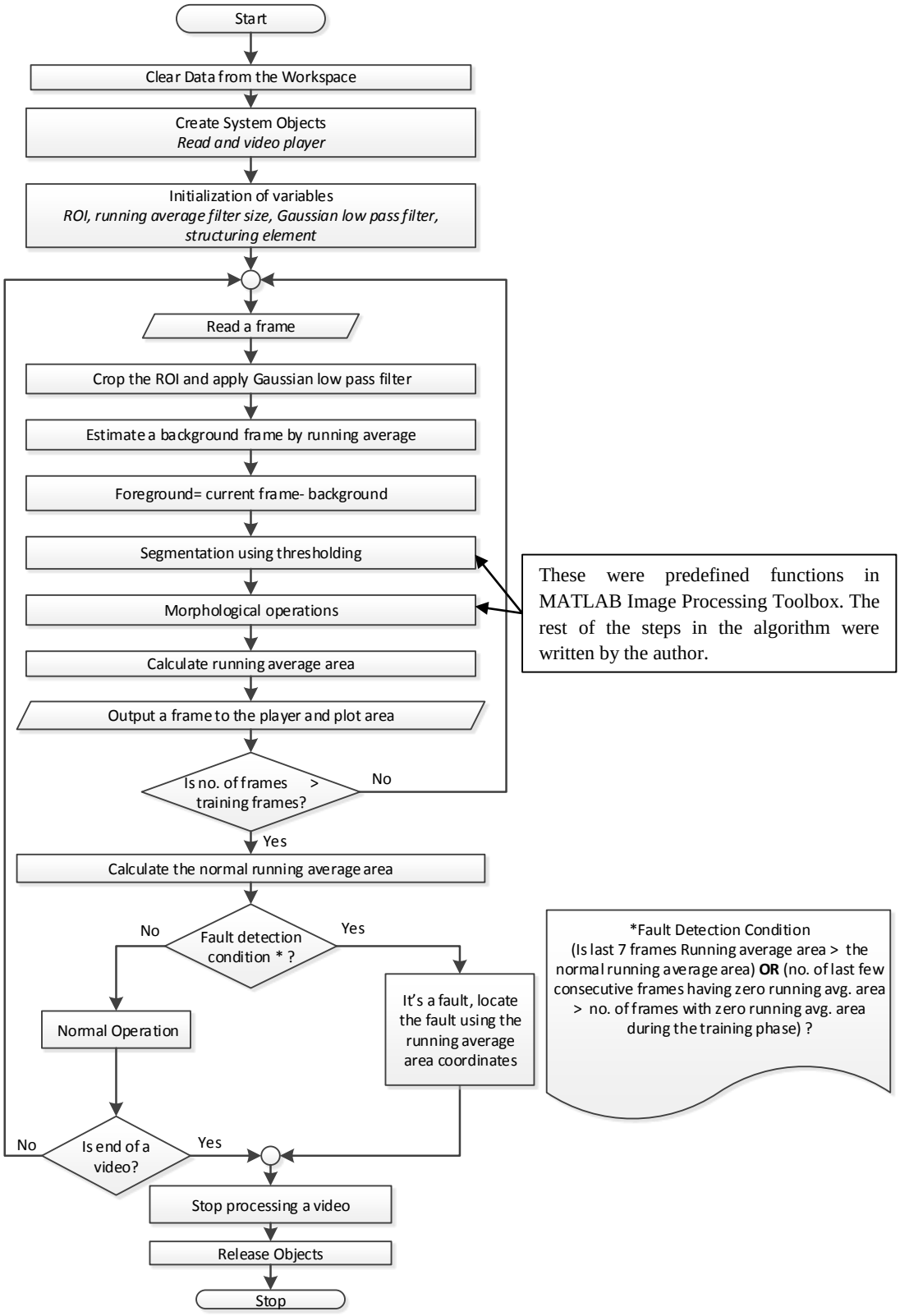


Figure 4-33 Flowchart for implementation of Method 3

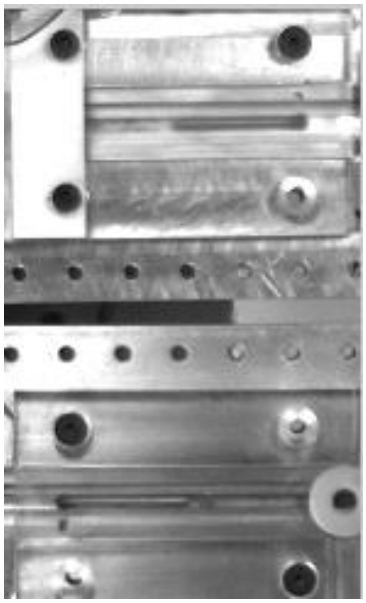


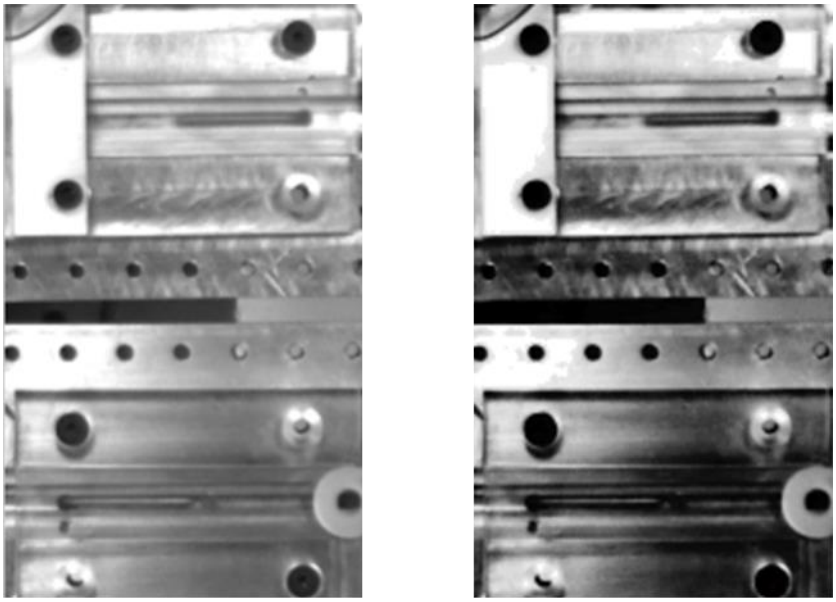
Figure 4-34 Cropped transfer track ROI to illustrate Method 3

The first step is to create a system object to read a video file. The video is read frame by frame and each frame is cropped to the ROI as defined by the user. For the transfer track region, the cropped ROI is shown in Figure 4-34.

Each image frame is processed with 5×5 Gaussian filter, as shown in Table 4-8, for noise removal. The filtering process reduces spatial noise but it blurs edges in the image as shown in Figure 4-35 (a). The next step is histogram equalization of the image to get uniform distribution of gray level values. Figure 4-35 (b) shows the image after histogram equalization. It can be seen that after histogram equalization, gray levels are redistributed and some of the white pixels are mapped to black and vice versa. A background frame is estimated from the last 20 frames of the video with the image averaging technique.

Table 4-8 5 × 5 Gaussian filter coefficients

0.0000	0.0000	0.0002	0.0000	0.0000
0.0000	0.0113	0.0837	0.0113	0.0000
0.0002	0.0837	0.6187	0.0837	0.0002
0.0000	0.0113	0.0837	0.0113	0.0000
0.0000	0.0000	0.0002	0.0000	0.0000



(a) (b)

Figure 4-35 (a) Filtered ROI with 5 × 5 Gaussian filter, (b) Image after histogram equalization

Figure 4-36 shows the estimated background frame from the last 20 frames with image averaging. Image averaging is a simple way of estimating the background. It may not give a result similar to GMMs, but it is fast and less computationally expensive. The background is subtracted from a frame to obtain the foreground image. Figure 4-37 (a) shows a frame and Figure 4-37 (b) shows the foreground after background subtraction. The foreground is grayscale and contains details of other objects along with the moving objects.

The foreground is converted into a binary image by segmentation with thresholding. The binary image contains moving objects with some noise. Moreover, the moving objects may not be full but they can be broken due to uneven lighting. A morphological *closing* operation with a disk shape structuring element of size 9 is applied to fill holes and smooth contours. The side effect of the closing operation is the results of narrow gaps. This is rectified by a morphological *opening* operation with a disk shape structuring element of size 3. The resulting image, as shown in Figure 4-38, is considerably noise free and it contains only moving object details. The resulting image has white pixels as assemblies and black pixels as the background. The ROI areas, T1 area and T2 area are calculated by counting the number of white pixels in the image. The area value is stored in a vector and the running average area of the last 7 frames is computed. Finally, a threshold (maximum blob area) is estimated for a normal operation from the first 200 frames.

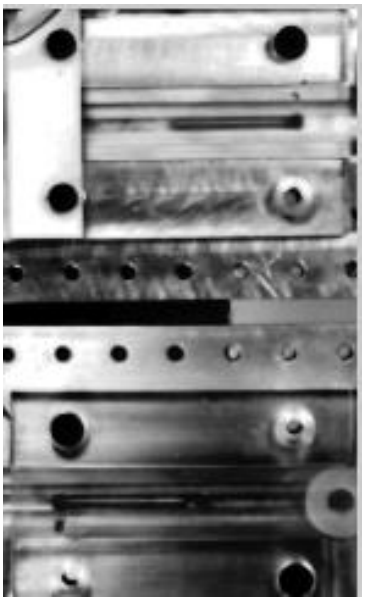


Figure 4-36 Estimated background frame with running average of last 20 frames

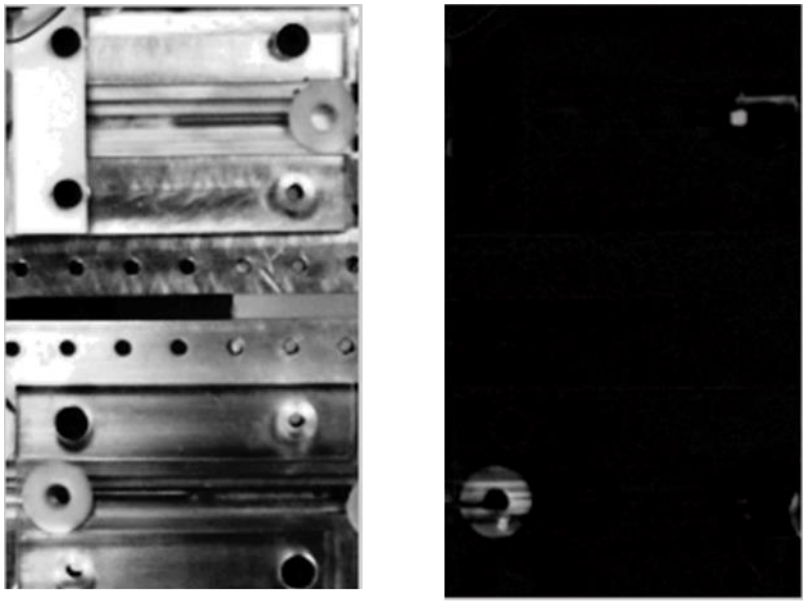


Figure 4-37 (a) A frame, (b) foreground from the frame after the background subtraction

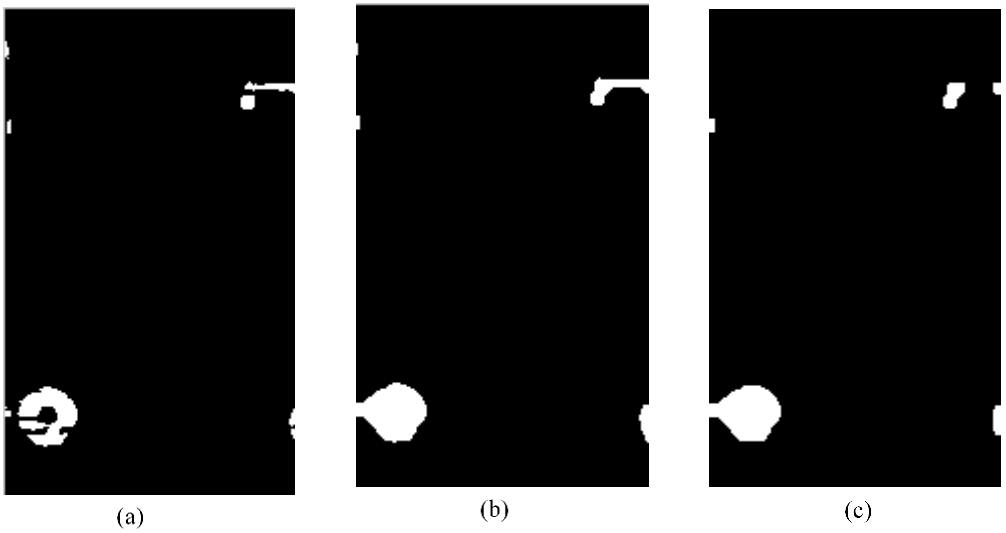


Figure 4-38 (a) Segmented binary image, (b) morphological closed image, (c) morphological open image

Figure 4-39 shows the plot of both the area and running average area for first the 200 frames (one full rotation of the wheels). The estimated threshold (maximum value of running average area from first 200 frames) is plotted as dotted line. The area trace is a noisy signal while the running average area trace is much smoother. In the next step, the current running average area is computed and compared with the threshold. If it stays within limits between zero and the threshold, the operation is classified as a normal. The steps are repeated till the end of a video file or a fault, whichever occurs first. Figure 4-40 depicts the plot of running average area vs number of frames for the normal operation through the transfer track region. It shows that for normal operation the running average area stays within the limits (0 and threshold).

If the area of the current frame exceeds the threshold, then Method 3 keeps track of the area for the next 7 consecutive frames. If the area of the subsequent frames keeps on increasing, then the condition is detected as a transfer jam. The limit of 7 consecutive frames is selected for fault confirmation because more than 7 frames added delay in fault detection while less than 7 frames misclassified normal operation as a jam. The jam is then classified as either transfer 1 or transfer 2 jam by locating the coordinates of the white pixels in the ROI. A GUI, similar to Method 1, was written in MATLAB for Method 3. The machine operating conditions are indicated by five indicators, one for a normal operation and four for the faults.

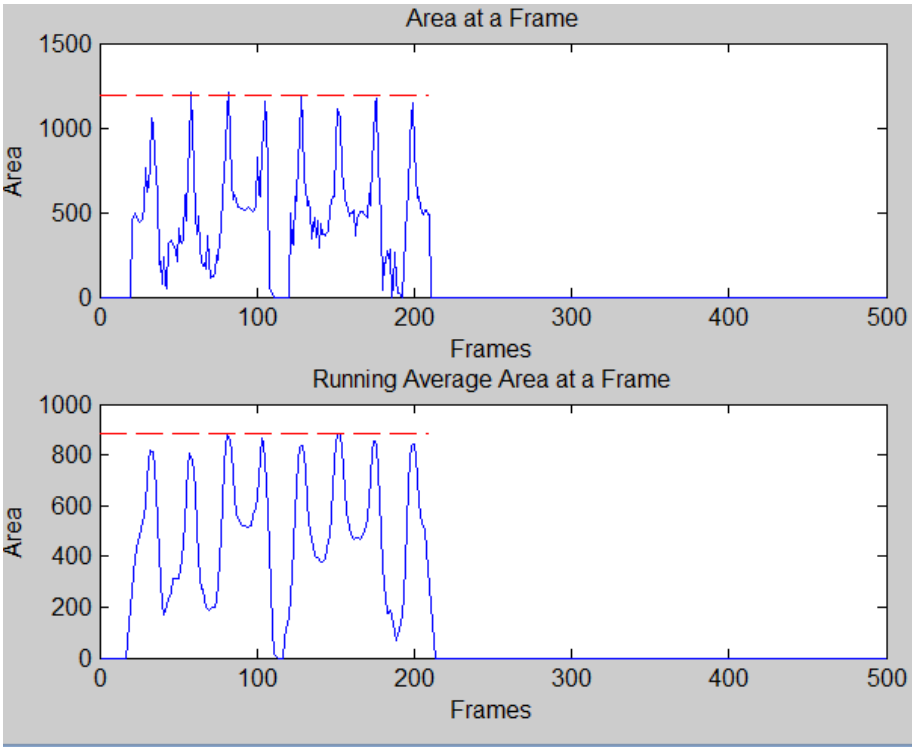


Figure 4-39 Running average area threshold estimation

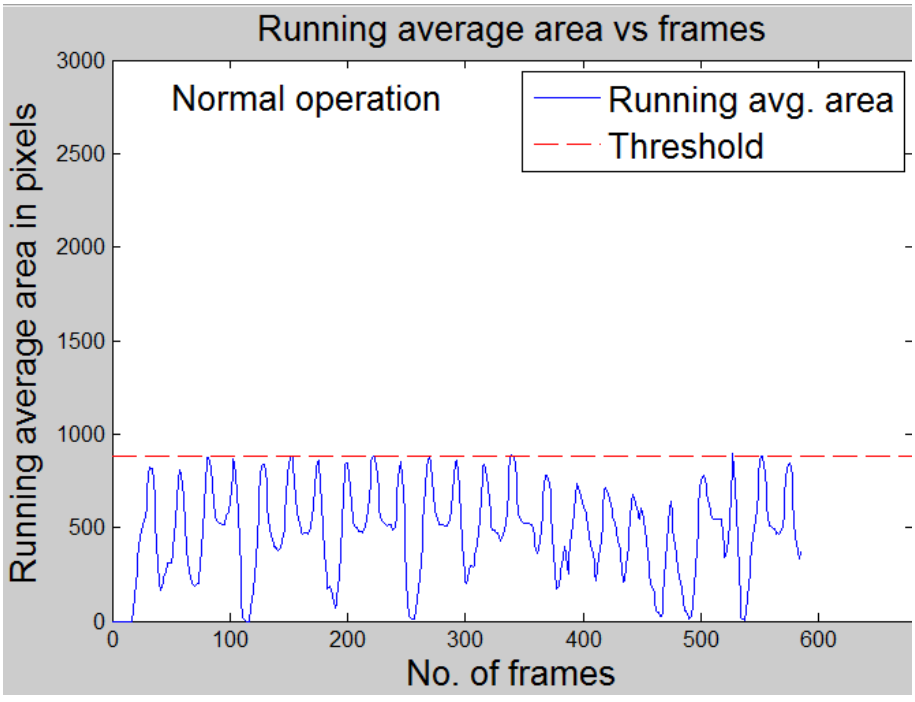


Figure 4-40 Running average area plot for a normal operation

Table 4-9 shows important events and fault signature for transfer 1 jam detection with Method 3.

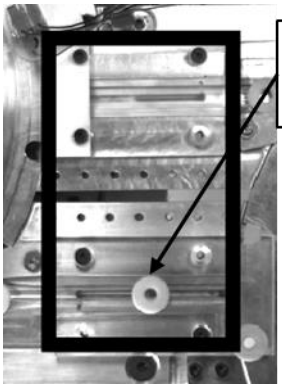
- Figure A: A carrier reached at the middle of the track in normal operation sequence.
- Figure B: The carrier was ready to engage in the slot of the secondary wheel. The fault was introduced using the HMI-PLC.
- Figure C: The carrier did not engage due to the absence of air pressure as a result of fault introduction. The running avg. area dropped due to lack of the motion of the carrier since Figure B.
- Figure D: Transfer jam fault detected by Method 3 and a high value (more than the threshold) of running avg. area was obtained due to more number of carriers in the track.
- Figure E: Continue running the machine past the fault resulted in the transfer track full of the carriers. The maximum value of running avg. area was obtained due to the presence of more carriers in the track.
- The above events are marked and shown in the fault signature plot. In practice, the method stopped processing the video file as soon it detected the fault. However, to get an idea of how bad a fault could have gone, the running avg. area till the end of video was plotted.

4.6 Summary

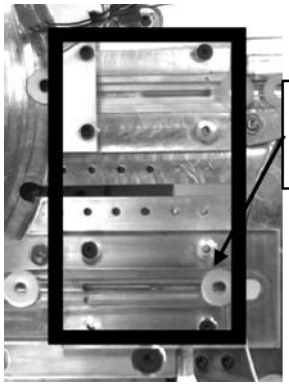
The methodology for acquiring video data for various operating conditions of the O-ring assembly machine was presented in the first section of the chapter. Two different cameras and software combinations were tested. Video data were acquired for preliminary testing with an IEEE 1394 camera and LabVIEW software. For the formal set of experiments, data were acquired with a high speed USB 3.0 camera and MATLAB software. Data acquisition was performed for both normal and faulty operating conditions of the machine. A total 35 video data sets were recorded for 3 normal and 4 faulty operating conditions corresponding to 3 ROIs. Three MVI methods were developed for detection and classification of the machine's operating conditions based on its video data sets.

Method 1 was based on GMMs and blob analysis for fault detection and classification. The method processed videos, extracted foreground objects with GMMs and calculated the foreground area as a feature with blob analysis. The normal operation behavior was learned with the first few initial frames of the video. The features were extracted from the next frames and compared with the normal operation features. If the values were within limits, the operation was considered as a normal operation for the given ROI. In the case of any deviations of the feature with respect to the normal value of the feature, the features were tracked for the next few frames to determine their trend. An increasing trend was classified as a fault and the fault was located with the coordinate values of the features

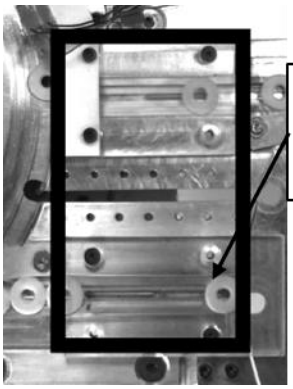
Table 4-9 Fault events and signature for transfer track 1 jam detection with Method 3



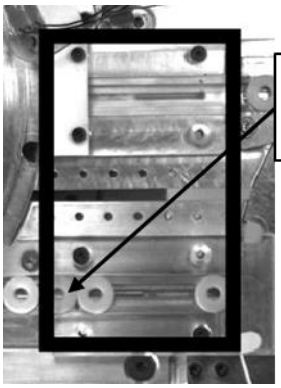
A: Normal frame (200th frame)



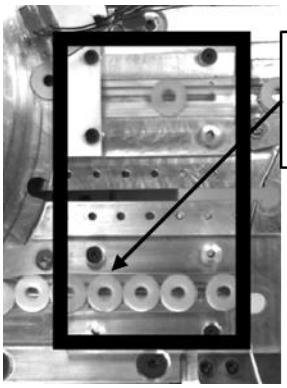
B: Fault introduction frame (300th frame)



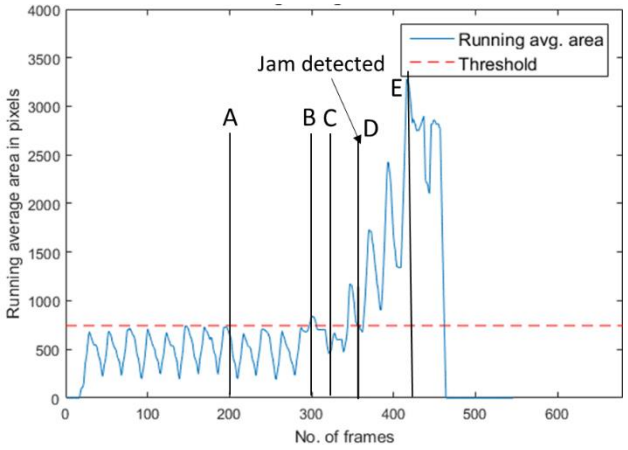
C: Fault visible frame (325th frame)



D: Fault detected frame (357th frame)



E: Fault's growth if undetected (430th frame)



Running avg. area plot for transfer track 1 jam

Method 2 used an optical flow technique for motion detection from a video to detect and classify the machine's operating conditions. Optical flow is a two dimensional vector that indicates the motion in x and y directions of a frame with respect to the previous frame. Using optical flow analysis, a feature called the OFD was calculated for each frame of the video. During normal operation, the OFD stayed within the limits. The limits were computed from a set of initial frames. The OFD was computed for every frame and its value was compared with the limits. An operation was classified as a normal operation if the OFD value was within the limits. If the OFD exceeded the limits, it was tracked and checked for the next few consecutive frames. If the OFD for the next few frames kept on increasing, the condition was classified as the faulty. The fault was located using the coordinates of the optical flow vector.

Method 3 was based on a running average technique for background detection, foreground estimation by background subtraction and morphological operations to detect a feature. The feature tracked was the running average area of the foreground pixels. The running average was used to reduce the effect of noise and to smooth the area feature value. A normal operation has the running average area within the limits. For each frame, the area was calculated and compared with the reference value obtained from the initial frames. If the running average area of a frame exceeded the limits and kept increasing, the machine condition was detected as a fault and the fault was classified based on the coordinates of the foreground pixels.

All three methods adopted the approach of learning normal behavior of the machine from the initial frames of a video. It is this observation that justifies identifying the methods as "intelligent". The limits for a feature of the normal operation were estimated and considered as the reference values. The limits were not fixed but instead they were dynamic in the sense that if there was a change in the lighting conditions or in the speed of the machines, the limits were changed online by the algorithm. The feature values were obtained from every frame and compared with the limits. The machine's operating condition was classified as faulty if the feature values exceeded the limits. The next chapter, presents the results for the experiments with the USB 3.0 camera and MATLAB combination.

Chapter 5

Results and Discussion

Fault detection and classification with a MVI system was carried out on an industrial O-ring assembly machine that assembles small circular O-rings onto continuously moving plastic carriers. The controlled faults were introduced with the HMI-PLC and video datasets were recorded for both normal and faulty operating conditions of the machine with a USB 3.0 digital camera. Video datasets were recorded for seven different conditions of the machine out of which three conditions were for normal operations and four conditions were for faulty operations. For each of these seven conditions, five video files were acquired and tested with the three MVI methods. The four faults on the machine were categorized into three ROIs namely transfer track ROI, air knife ROI and the hopper ROI. Time was recoded, in terms of a frame number, every time a controlled fault was introduced. The time was then used to compare the performance of three MVI methods in terms of how fast each method detected a fault. The time taken to process a 10s long video file (300 frames) by each method was measured and recorded for the performance comparison. All algorithms were run on a desktop PC with a configuration, Intel® Core ™ i7-2600 CPU @ 3.40 GHz, 8 GM RAM and 64-bit operating system.

Each MVI method needed a certain number of frames to "see" a fault after it was introduced. The number of frames varies between methods. The delay was measured by recording the frame number at the time of fault detection by each method. The frame number for when the fault was initiated by the HMI-PLC was also recorded. An algorithm's processing time to handle 10s long video file depended on the complexity of the algorithm. For example Method 3 took more time to process the video files because of the computational time required to implement the running average filters. It was also observed that all algorithms took more time to process when the frames were displayed in real time as they were being processed. It is well known that a MATLAB frame display adds significant delay in processing time. Due to the communication lag in the video file acquisition procedure, all video files were not of the same length. Hence, the time taken to process a complete video file was also not the same. The average time taken to process the video dataset was calculated and used for the performance measurement of each MVI method. Section 5.1, 5.2 and 5.3 discuss results with Method 1, Method 2 and Method 3, respectively. Performance comparison between the three MVI methods for fault detection is discussed in Sections 5.4 and 5.5. Section 5.6 summarizes the chapter.

5.1 Method 1: Fault Detection with GMMs and Blob Analysis

Method 1 used foreground estimation with GMMs followed by blob analysis and feature extraction for fault detection and classification on the O-ring assembly machine. The blob area was measured for each frame and plotted w.r.t. number of frames. The blob area for a normal operation cycle was within the limits calculated from the first few initial training frames. The method was tested for each of the three ROIs and performance parameters were tabulated. The method was less sensitive to variation in lighting as it updated GMMs parameters dynamically during processing. The method was tested with all 35 video files acquired for seven different operation conditions of the machine. But as an example only one of the files, as listed in Table 5-1, from each of the seven conditions are presented and explained in this section. The results are presented by ROI, that is by transfer track ROI, air knife ROI and the hopper ROI. For each ROI, the results are given for all operating conditions within that region. In the case of the transfer track ROI, this includes 3 conditions: normal operation, transfer 1 jam and transfer 2 jam. The single air knife fault and normal operation of the air knife are given for the air knife ROI. Similarly, hopper normal operation and the single hopper fault are presented for the hopper ROI.

5.1.1 Transfer Track ROI Results with Method 1

The transfer track region covers three operating conditions: normal operation, transfer track 1 jam and transfer track 2 jam. The ROI was fixed once the camera was mounted at the appropriate working distance. The ROI covers both the transfer 1 and transfer 2 tracks. Due to non-uniform lighting and shadows from the column, transfer track 1 looks brighter compared to transfer track 2. One example result is presented for a normal operation (NO5), one for a transfer 1 jam (T1JAM1) and one for a transfer 2 jam (T2JAM3). The full blob area plot, until the end of a video file, is plotted for normal operation. For the transfer jams, the algorithm stopped processing the video files as soon it detected the fault in order to demonstrate that the fault was detected before it became severe.

A sample image and plot for normal operation through the transfer track region are shown in Figure 5-1 and Figure 5-2 respectively. The ROI is shown as a rectangle in the frame. The foreground blob area was measured in pixels and plotted w.r.t. the frames as a solid line in the plot. The estimated threshold from the first few training frames was plotted as a dashed line in the plot.

Table 5-1 Video files considered for discussion of results

	Transfer track ROI	**Air knife ROI**	**Hopper ROI**
Normal operation	NO5	AirNO1	Hopper NO5
Faulty Operation	T1JAM1 and T2JAM3	Air knife fault 1	Hopper fault 2

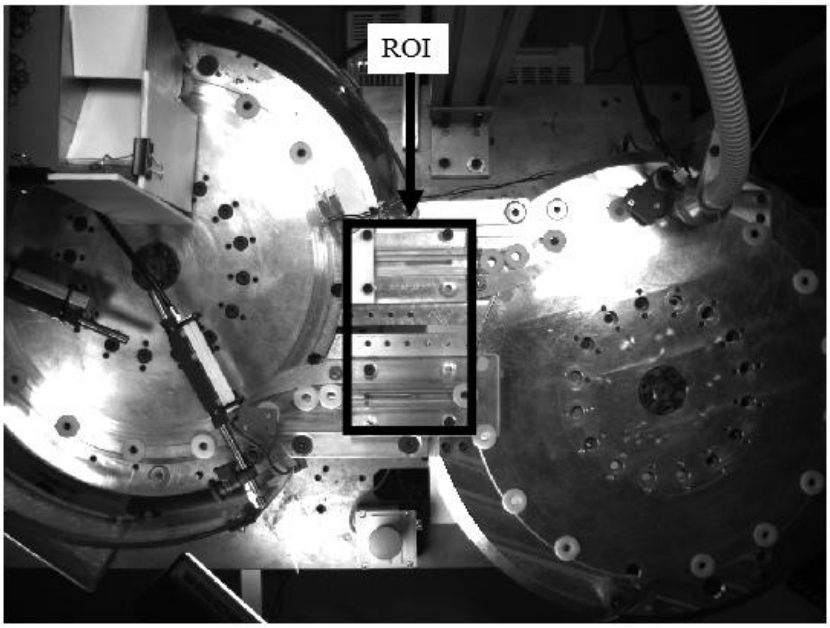


Figure 5-1 A transfer track normal operation frame with ROI labelled

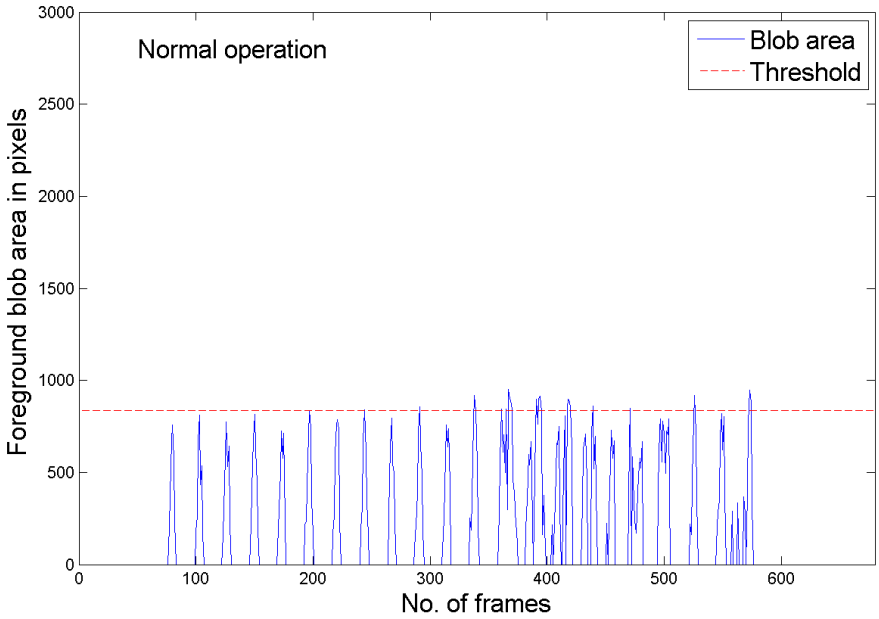


Figure 5-2 A transfer track plot for normal operation using Method 1

The carriers move through the transfer track region at a speed of 0.6 m/s. The passage in the transfer track region is narrow to prevent air leaks. Hence, if a carrier is not aligned properly in the passage or if the friction is high, it will result in a blockage in the transfer track. Sometimes, the blockage is temporary in the sense that one carrier blocks the track but the following carrier pushes it and recovers the track from a temporary jam. Method 1 measures a blob area for each frame in pixels and estimates the threshold for a normal operation cycle. It can be seen in Figure 5-2 that the blob area is not constant w.r.t number of frames; this is due to non-uniform flow of the carriers through the ROI as a consequence of fluctuations in the air pressure and friction. The frame rate of video acquisition was higher than the rate of movement of carriers. Therefore, not all frames had carriers visible in them. Some frames have only background and no carriers in them. Since Method 1 measures only foreground area, it does not plot any value for the background. Hence, the blob area for frames with missing carriers is zero. At some point, the blob area in a particular frame exceeds the threshold but it is only for a single frame due to a temporary jam in the track. The algorithm is trained not to classify this false jam as a transfer jam.

A typical transfer track 1 jam image and plot are shown in Figure 5-3 and Figure 5-4, respectively. The jam was introduced at frame no. 300 and it was detected at frame no. 348 with a delay of 48 frames. The plot shows the estimated threshold as a dashed line as generated from the first few training frames. At some point, the blob area for a frame exceeds the threshold but it returns within the limits upon recovery from a temporary jam. Once a jam is detected, the algorithm keeps track of it for the next 5 consecutive frames and if the jam increases w.r.t. time, it is considered as an actual jam. In the event of jam, the jam indicator is set red as an alarm and the method stops further processing of the video file. The full plot is given in Figure 5-4 to show that the jam could have been serious if the method had not detected the fault. The bounding box for the jam is shown as the rectangle on the transfer track region.

Figure 5-5 and Figure 5-6 presents results for a transfer 2 jam. The transfer track 2 jam was introduced at frame no. 320 and it was detected at frame no. 343 with a delay of 23 frames. It can be seen that the foreground blob area stayed within the limits until the fault was introduced. The fault was classified with the coordinates of the bounding box and it is shown in the circle in the image. Method 1 took on average 3.42 s to process a 10 s transfer track video files with the frame display command active in the algorithm. Method 1 was applied to all 15 video files and no False Positive (FP) and False Negative (FN) results were reported. Thus, the accuracy was 100 %. However, there was a delay between the fault introduction and its detection by the method. It took nearly 36 frames on average to classify the condition as a fault after it was initiated. This is equivalent to 1.2 s delay in fault detection. However, this delay was not a concern as it can provide a time for the MVI system to stop the machine and prevent any damage.

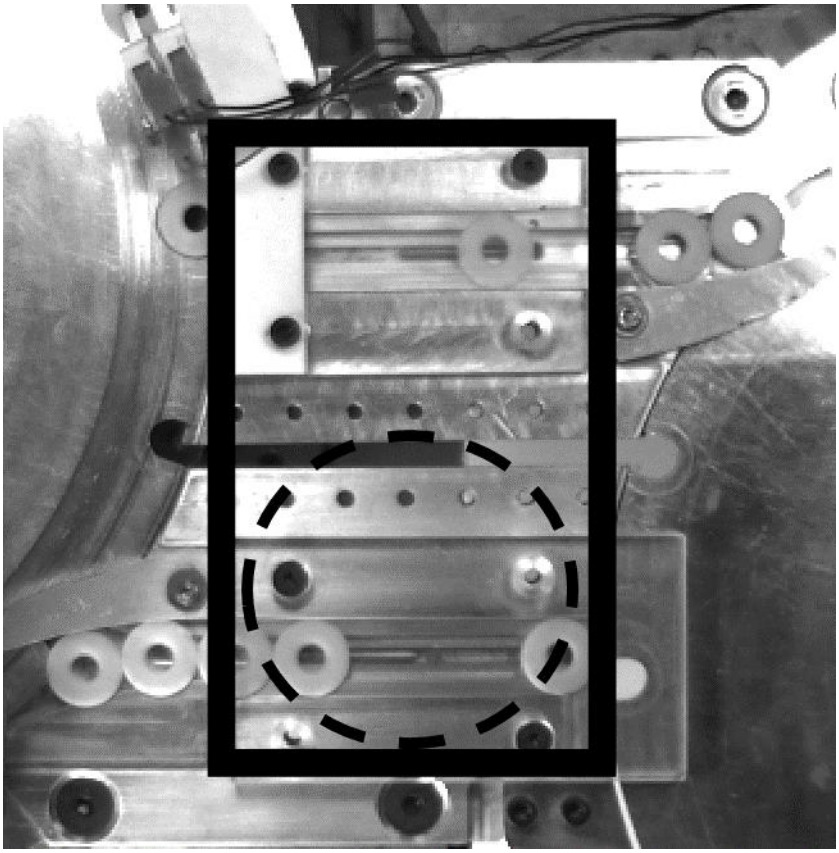


Figure 5-3 A transfer 1 jam frame with fault circled within the ROI

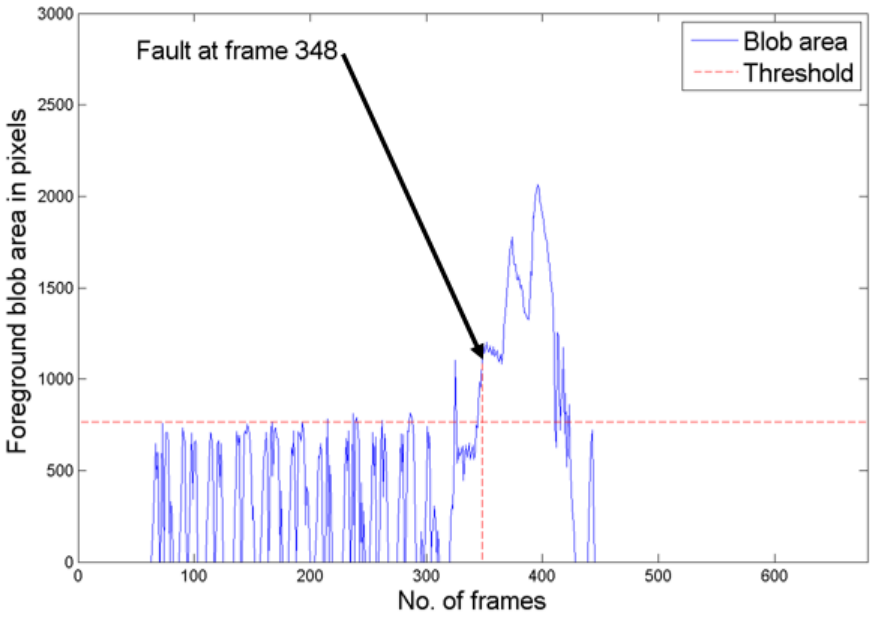


Figure 5-4 A transfer 1 jam plot using Method 1 with fault detection frame identified

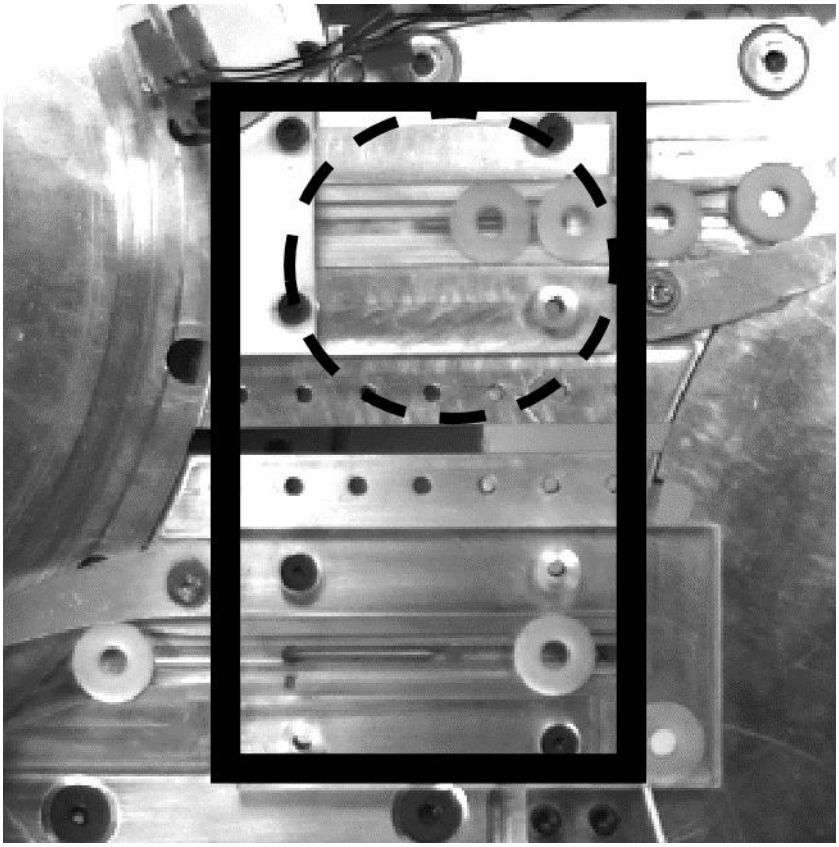


Figure 5-5 A transfer track 2 jam frame with fault circled within the ROI

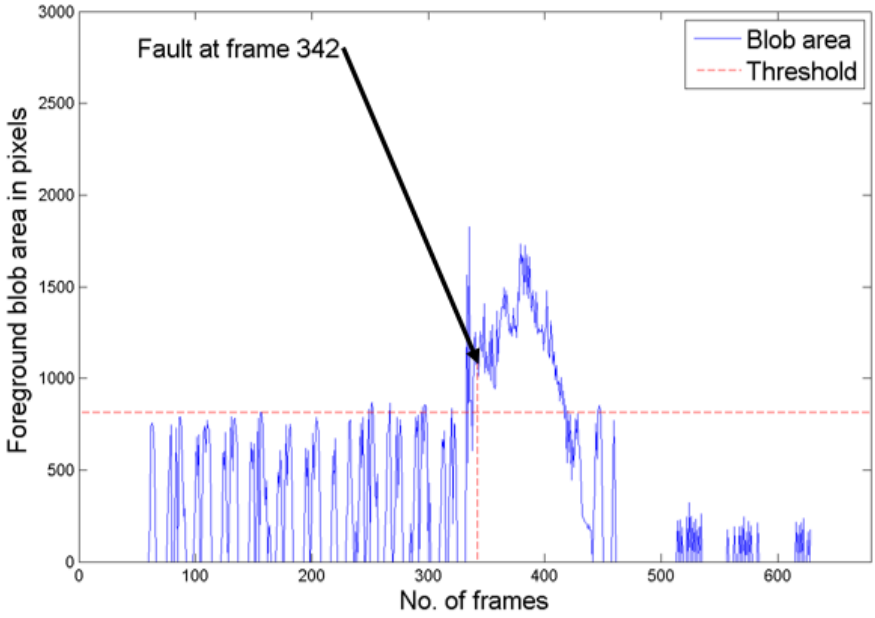


Figure 5-6 A transfer track 2 jam plot using Method 1 with fault detection frame identified

5.1.2 Air Knife ROI Results with Method 1

The nature of an air knife fault is very different from that of a transfer track fault. In the case of a transfer track fault, the carriers move through the ROI constantly and in the event of a jam, more carriers are present in the track than during normal flow. Hence, Method 1 detects a high foreground area for the blobs in the case of a transfer jam. The function of the air knives is to remove excess unassembled O-rings from the primary wheel and blow them into the collection vacuum bin. Hence, the air knife ROI has the absence of O-rings in the region for a normal operation. In case of a fault (i.e. when air knives do not work as intended), O-rings are present in the air knife ROI. Thus, for the air knife fault, the ideal foreground blob area should be zero for normal operation and it should be high for a fault. However, under practical conditions, air knives are not able to remove all excess unassembled O-rings from the primary wheel. A few O-rings will be present in the air knife region even during normal operation. Therefore, the foreground blob area for normal operation for the air knife ROI may have random non-zero values for some frames due to the presence of a small number O-rings.

Figure 5-7 and Figure 5-8 show a sample frame and the foreground blob area plot for normal operation through the air knife ROI. The ROI is shown as the square box in the image. For normal operation, this region should be free from unassembled O-rings. The threshold for the normal operation should be zero but in practice there are some scattered O-rings present on the primary wheel at the air knife region. Hence, the threshold cannot be zero. The method calculates the threshold by measuring the foreground blob area from a few initial training frames. The plot shows the threshold estimated from the training frames as a dashed line. As long as the foreground blob area in the ROI stays under the threshold or if it exceeds the threshold for less than 5 frames, the operation is classified as normal operation.

Figure 5-9 and Figure 5-10 show an air knife fault frame and the corresponding foreground blob area plot for the air knife ROI. The presence of O-rings can be seen inside the circle on the primary wheel. These are excess O-rings that might cause a transfer track jam if they enter in the transfer track region along with the carriers. The O-rings move through the air knife region at a speed of 0.089 m/s. For a faulty video file, the threshold of 300 pixels is obtained from the training frames. The fault was introduced at frame no. 290 and it was detected at frame no. 313 with a delay of 23 frames. Method 1 took on average 3.10 s to process a 10 s air knife video dataset with the code executed in MATLAB. Method 1 was applied to all 10 air knife region video files and no FP and FN results were reported. The method had successfully classified all air knife video files into their correct categories. Thus, the accuracy was 100%.

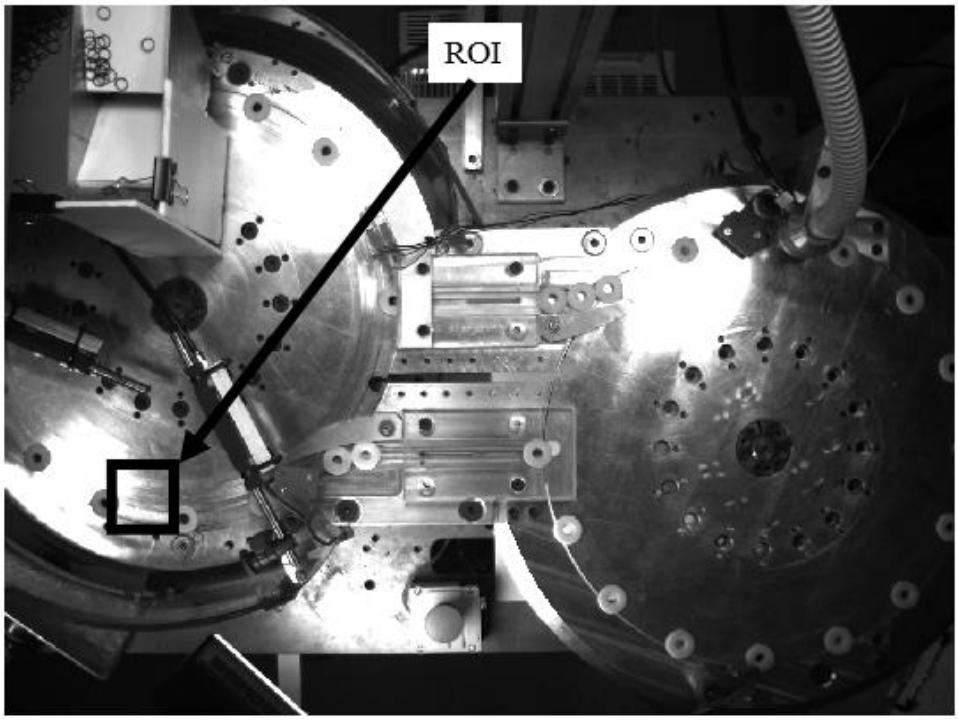


Figure 5-7 An air knife normal operation frame with ROI labelled

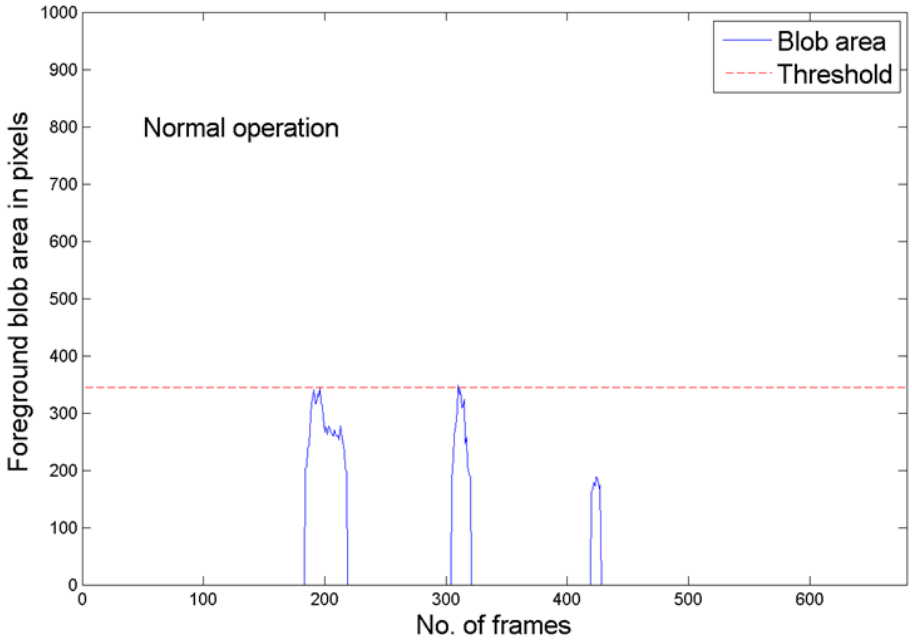


Figure 5-8 An air knife normal operation plot using Method 1

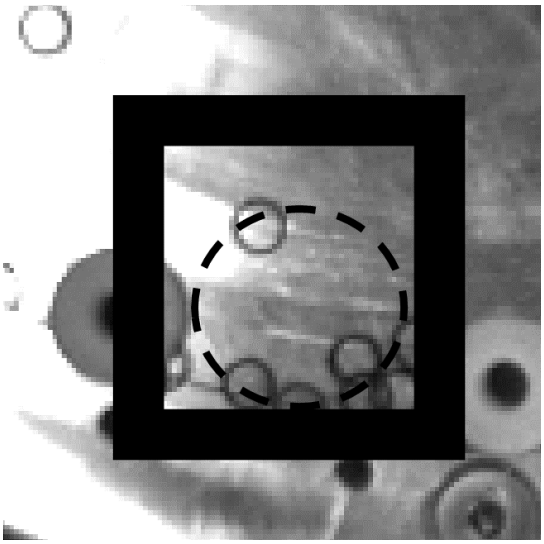


Figure 5-9 An air knife fault frame with fault circled within the ROI

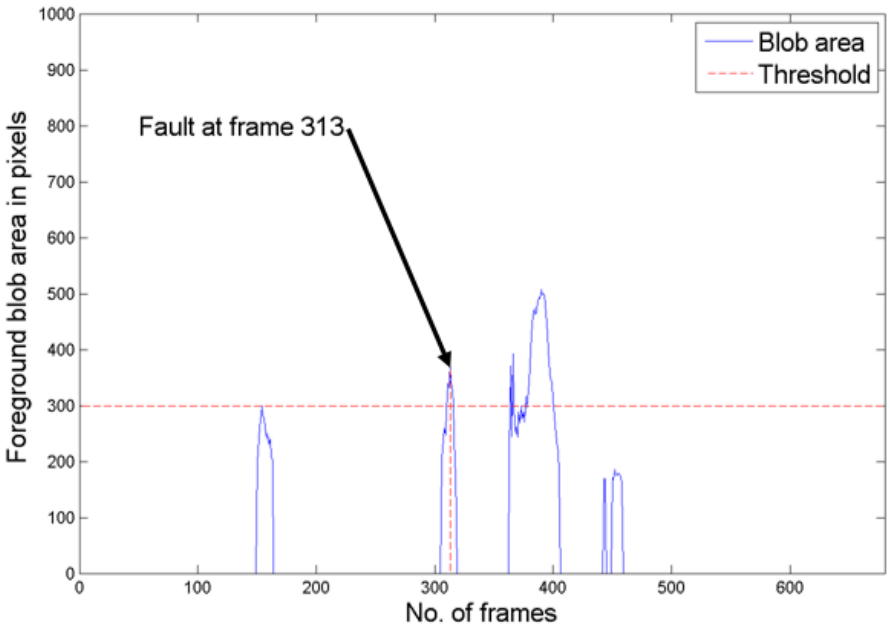


Figure 5-10 An air knife fault plot using Method 1 with fault detection frame identified

5.1.3 Hopper ROI Results with Method 1

The nature of a hopper fault is opposite to that of an air knife fault in two ways. The first difference is in terms of the foreground objects and the background; the objects are on a moving background for the air knife ROI, while the objects are on a stationary background for the hopper ROI. The second difference is that in case of the air knife fault, the O-rings are absent during a normal cycle. By contrast O-rings are present during the normal cycle for a hopper fault. In other words, for an air knife fault, more O-rings are present in the ROI indicating a lack of air pressure, while for a hopper fault O-rings are absent in the ROI indicating a lack of O-ring flow from the hopper. For the air knife fault, a high foreground blob area is the indication of the fault, while for the hopper fault a low foreground blob area is the indication of the fault. Therefore, ideally the foreground blob area for a normal operation of the hopper ROI should have a high value and a faulty operation should have zero value, but the blob area may have random values for some frames due to non-continuous flow of the O-rings.

Figure 5-11and Figure 5-12 show the sample frame and the foreground blob area plot for the normal operation through the hopper ROI. The ROI is shown as the square box in the image. For normal operation, this region should have a few O-rings in all frames. The threshold value for the normal operation should be high, but in practice there are some frames without O-rings due to non-uniform flow in the hopper. The method calculates the threshold by measuring the foreground blob area from a set of initial training frames. The method tracks the number of frames having zero or low blob area during the training phase. The plot shows the threshold estimated from the training frames as a dashed line. The effect of non-uniform flow of the O-rings can be seen as the fluctuations in the foreground blob area. This is due to friction in the hopper and the inability of the vibrator to give a steady flow. O-rings from the hopper fall onto the tray and pile on to each other. They move a few mm distance with the vibration but they fall into the feed chute only when they reach the end of the tray. Sometimes, they pile up and it takes time for the “log jam” to break and for the flow to resume. As long as the foreground blob area in the ROI stays within the threshold and the number of frames having zero or low blob area stays less than the estimated range from the training frame, the operation is classified as normal operation.

Figure 5-13 and Figure 5-14 show a hopper fault frame and the corresponding foreground blob area plot for the hopper ROI. The absence of O-rings is circled in the feed chute. The O-rings move through the hopper region at a speed of 0.32 m/s. The fault was introduced at frame no. 230 and it was detected at frame no. 266 with a delay of 66 frames. Method 1 took on average 3.43 s to process a 10 s long hopper region video file. Method 1 was applied to all 10 hopper region video files and it successfully classified all the files into their correct categories. Thus, once again it achieved an accuracy of 100 %.

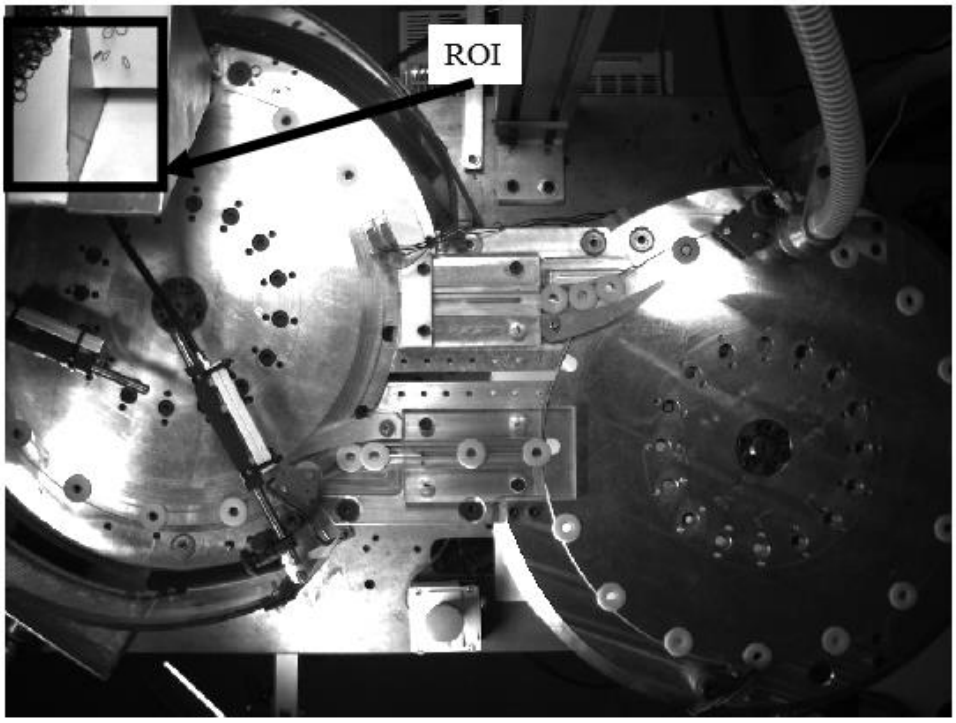


Figure 5-11 A hopper normal operation frame with ROI labelled

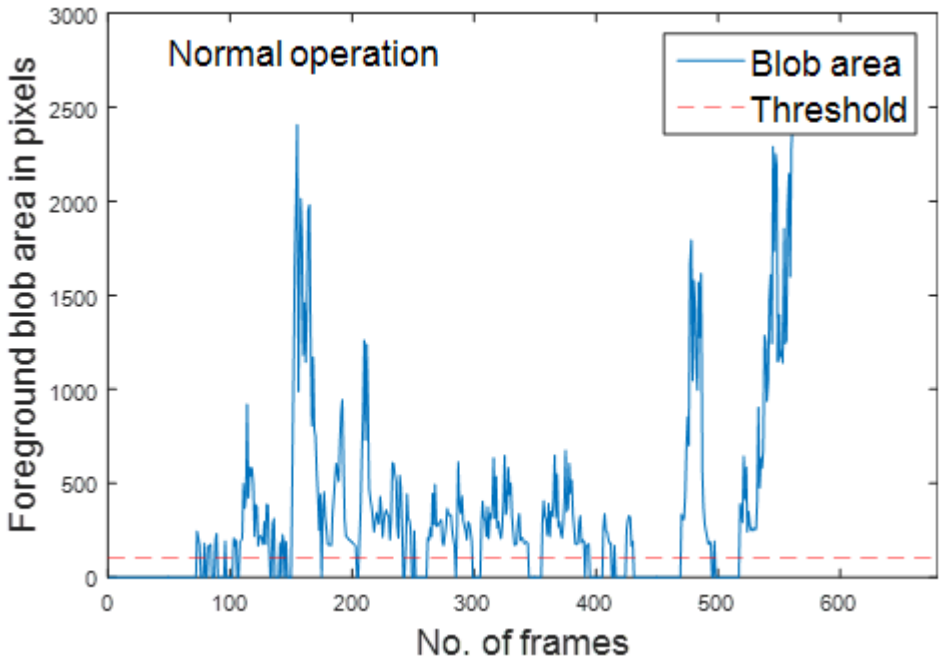


Figure 5-12 A hopper normal operation plot using Method 1

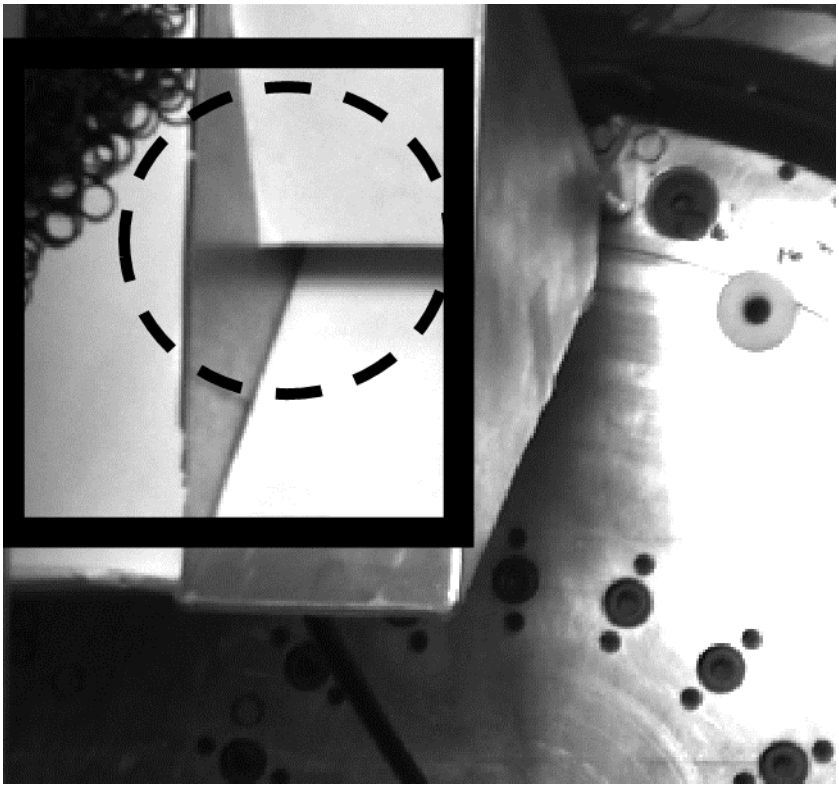


Figure 5-13 A hopper fault frame with fault circled within the ROI

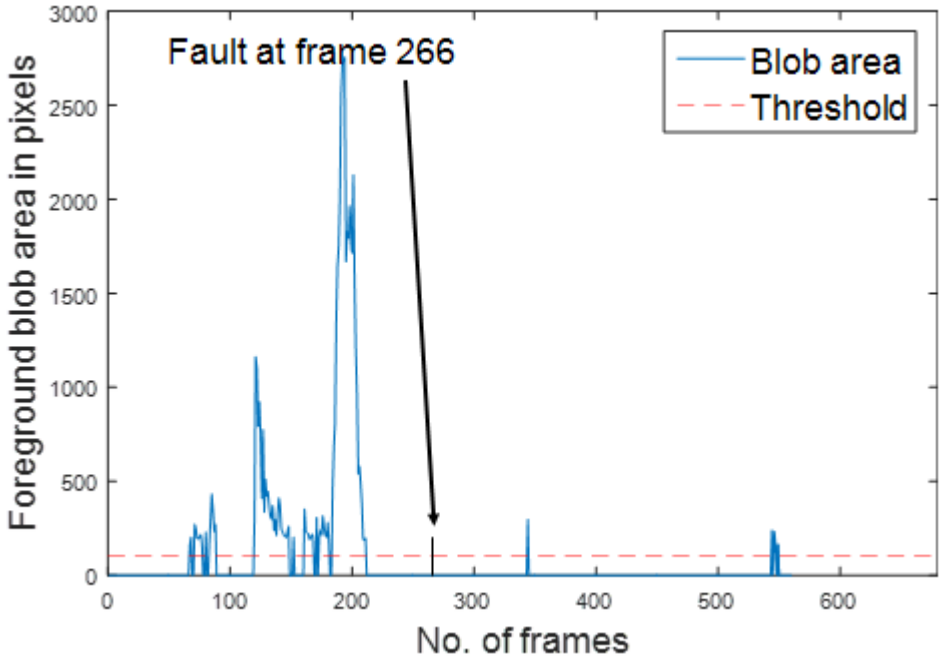


Figure 5-14 A hopper fault plot using Method 1 with fault detection frame identified

5.2 Method 2: Fault Detection with Optical Flow Estimation

Method 2 uses optical flow for motion detection and fault classification from a video. Optical flow is a basic motion detection technique for video analysis. It calculates the optical flow for an operation sequence, which is the measure of movements of foreground objects. The optical flow is a two dimensional vector which is the measure of the motion in the X and Y directions. The optical flow is estimated with the Horn-Schunck method and it is a complex number. The Optical Flow Density (OFD) was computed from the optical flow vectors of the video file frame. The OFD, the sum of the absolute values of optical flow vectors, is a single number that represents the entire motion in a frame. The optical flow is very sensitive to small scale motions such as vibrations on the machine or fluctuations in the light. The estimated optical flow was subsampled and scaled to minimize the effect of minor motions. During normal operation, the OFD stayed within the limits. The OFD for normal operation was considered as the reference value. For each frame, the OFD was determined and compared with the reference (threshold) value. If the current frame OFD exceeded the reference value and kept on increasing with respect to time, then that was the indication of large scale motion in the frame. The method then kept track of the OFD for the next 5 frames and if the trend continued, the machine condition was considered as a fault, otherwise it was judged as a normal operation.

5.2.1 Transfer Track ROI Results with Method 2

The transfer track region has three operating conditions: a normal operation through the transfer track, transfer 1 track 1 jam and transfer track 2 jam. The same video files as used in Method 1 are used as the examples for the fault detection with the Method 2. For all examples, a sample frame image with the corresponding OFD vs frames plots are presented and discussed. Figure 5-15 and Figure 5-16 indicate the image and plot for a transfer track normal operation. The sensitivity of the optical flow can be seen as the presence of noise in the OFD plot. The threshold value of the OFD was estimated with the initial training frames and it is plotted as a dashed line in the plot. The estimated threshold value of the OFD for the normal operation cycle is 38 pixels for this normal operation video file. The OFD from each frame is calculated and compared with the threshold. It can be seen that for a number of frames, the OFD exceeds the threshold but it does not increase with time and it returns to its normal value within a few consecutive frames. Hence, the operation was classified as normal. Method 2 was applied to all 15 video files for transfer track region and 4 FP and 4 FN results were reported. Thus, the accuracy was 47 %. The method interpreted the 4 operations as FN because it considered the noise in the normal operation video file as the high value of the OFD. Method 2 took on average 3.83 s to process a 10 s long video dataset.

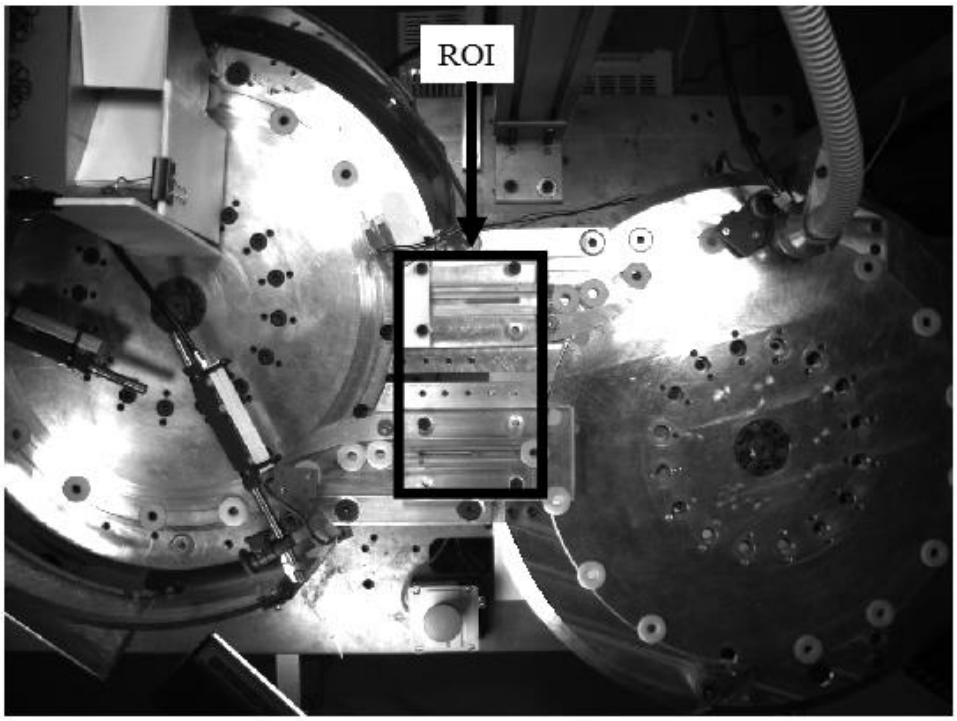


Figure 5-15 A transfer track normal operation with ROI labelled (same as Figure 5-1, repeated for consistency)

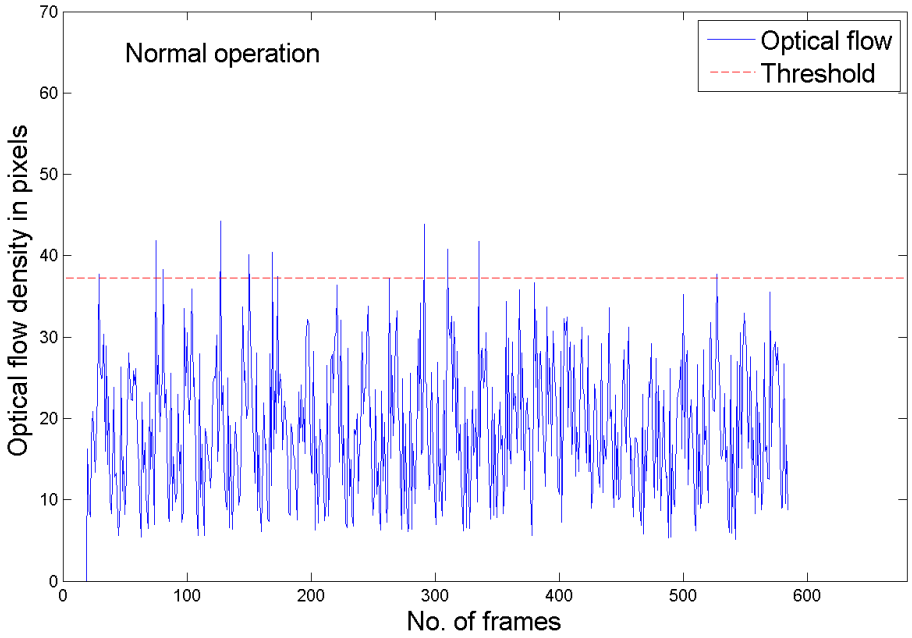


Figure 5-16 A transfer track normal operation plot using Method 2

Transfer track 1 jam detection with Method 2 is shown in Figure 5-17 and Figure 5-18. The fault is circled in Figure 5-17 and the optical flow lines can be seen as short yellow lines on the carriers. They also appear on the track due to noise. The fault was introduced at frame no. 300 and it was detected at frame no. 420 with a delay of 120 frames. However, the speed of processing slows down substantially if the ROI is increased in size.

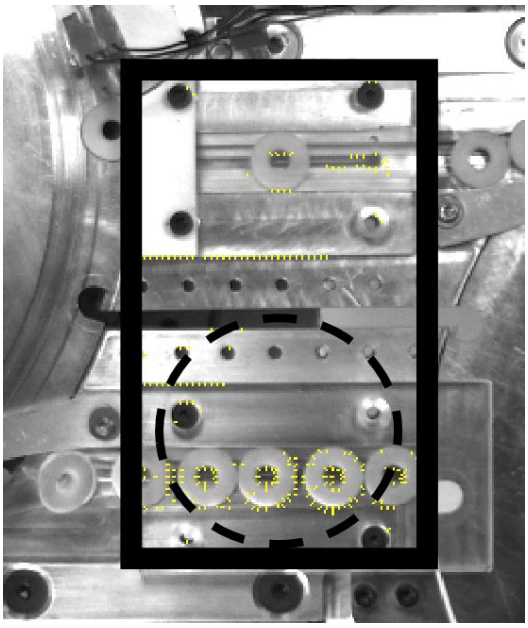


Figure 5-17 A transfer track 1 jam frame with fault circled and optical flow lines in yellow

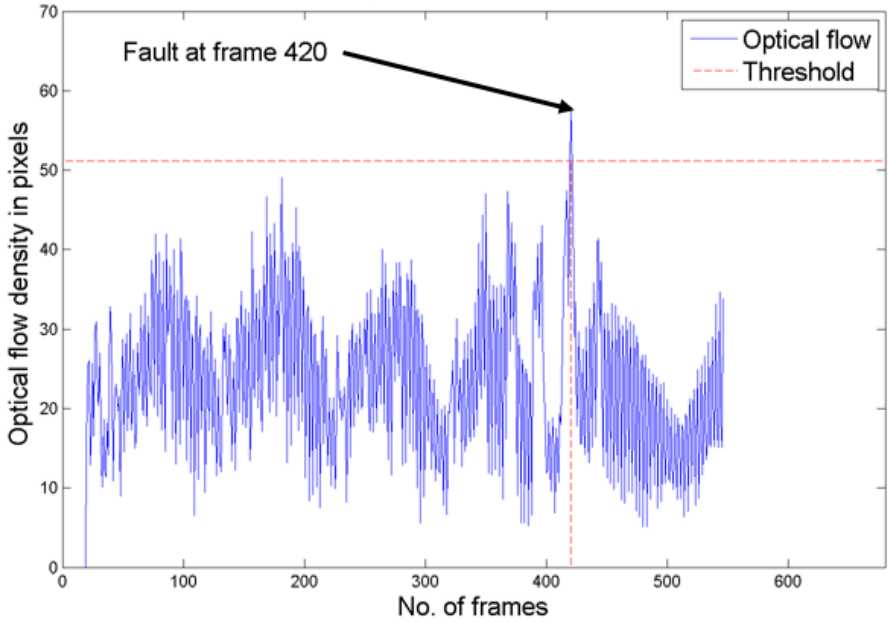


Figure 5-18 A transfer track 1 jam plot using Method 2 with fault detection frame identified

Transfer track 2 jam detection with Method 2 is shown in Figure 5-19 and Figure 5-20. The fault is circled in Figure 5-19 and the optical flow lines can be seen as short lines on the carriers. The fault was introduced at frame no. 320 and it was detected at frame no. 333 with a delay of 13 frames.

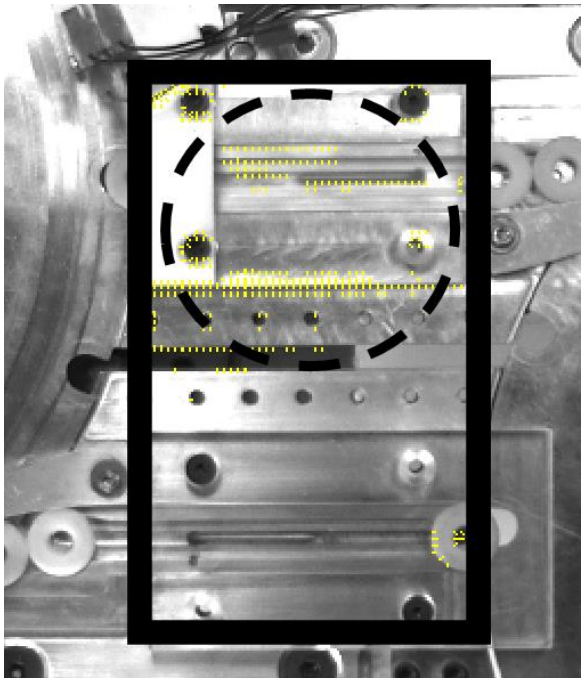


Figure 5-19 A transfer track 2 jam frame with fault circled within the ROI and optical flow lines visible

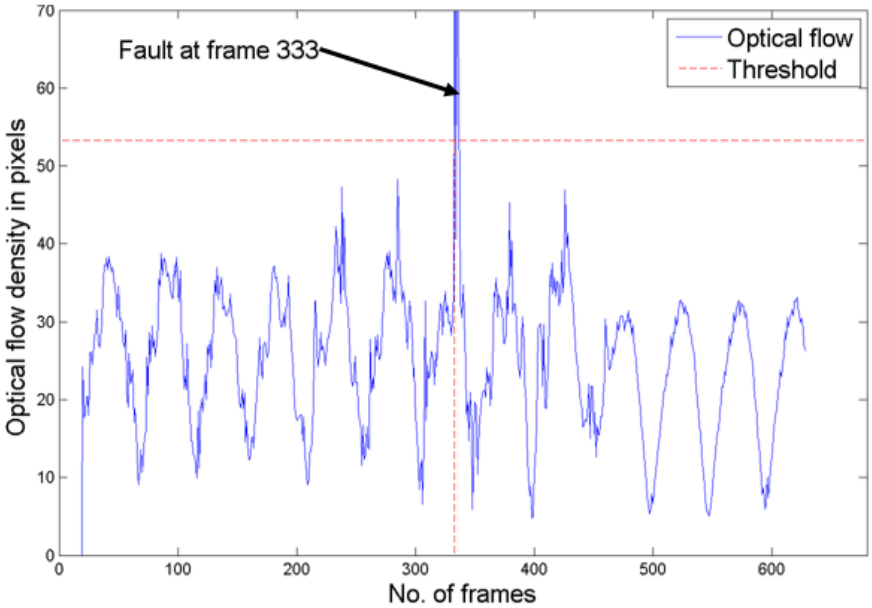


Figure 5-20 A transfer track 2 jam plot using Method 2 with fault detection frame identified

5.2.2 Air Knife ROI Results with Method 2

Air knife normal operation detection with the Method 2 is shown in Figure 5-21 and Figure 5-22. The normal operation is indicated inside the square box. The OFD for a normal operation is below the estimated threshold as shown in the plot. Method 2 took on average 3.10 s to process air knife region video dataset.

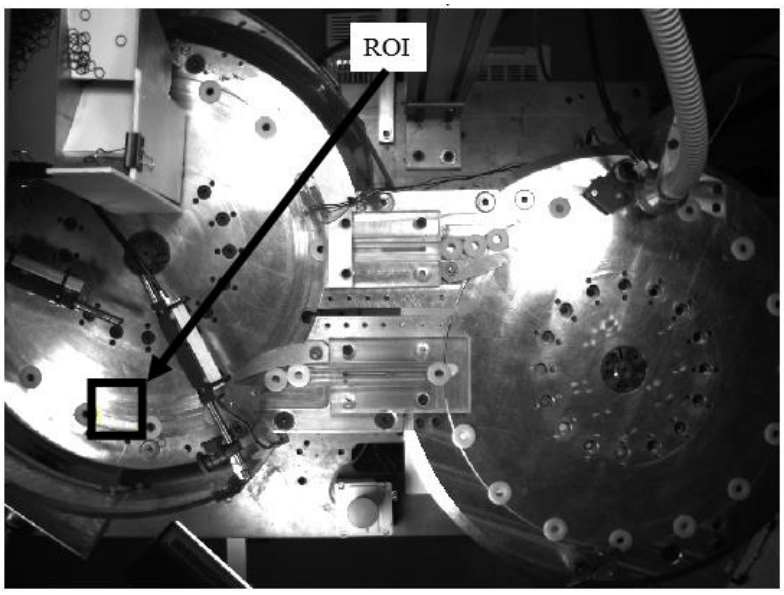


Figure 5-21 An air knife normal operation frame with ROI labelled (same as Figure 5-7, repeated for consistency)

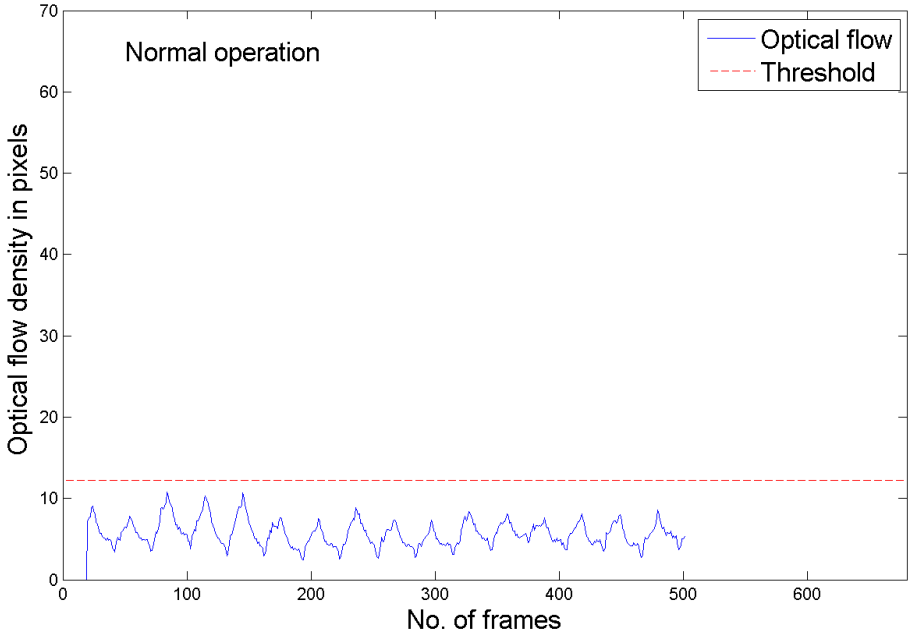


Figure 5-22 An air knife normal plot using Method 2

Air knife fault detection with Method 2 is shown in Figure 5-23 and Figure 5-24. The fault is circled and the optical flow lines can be seen as short lines on the O-rings. The fault was introduced at frame no. 290 and it was detected at frame no. 310 with a delay of 20 frames. Method 2 applied to air knife video dataset resulted in 10 out of 10 correct classifications. Thus, the accuracy was 100 %.

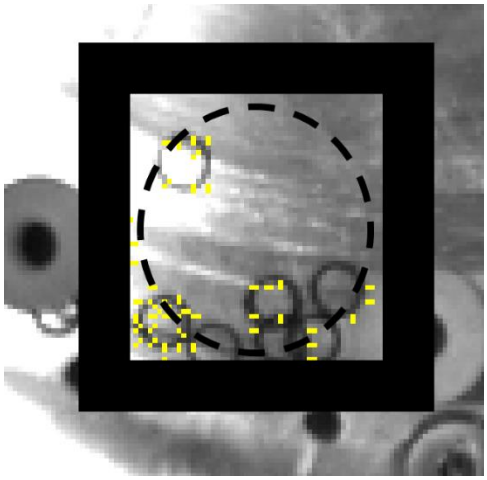


Figure 5-23 An air knife fault frame with fault circled within the ROI

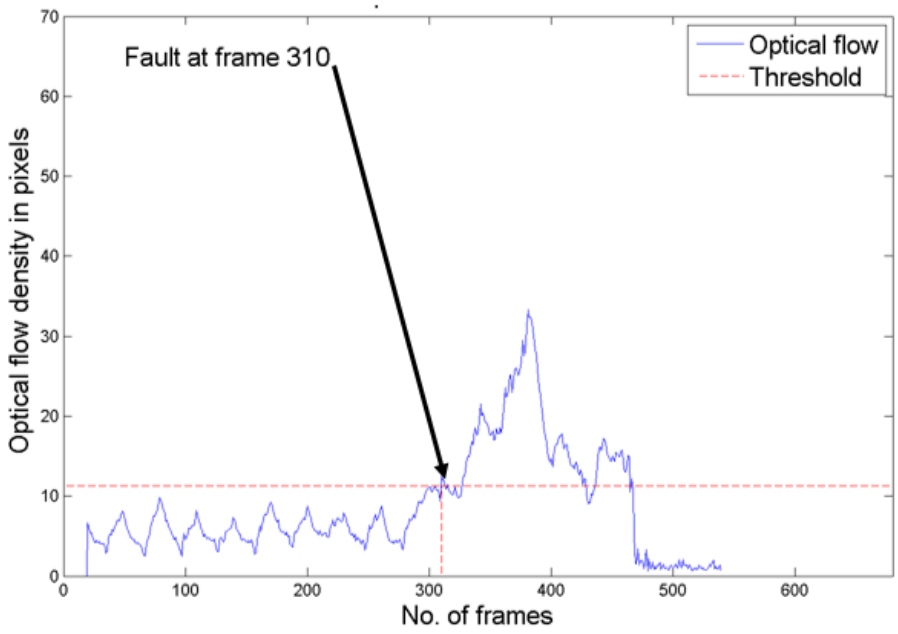


Figure 5-24 An air knife fault plot using Method 2 with fault detection frame identified

5.2.3 Hopper ROI Results with Method 2

Hopper normal operation detection with Method 2 is shown in Figure 5-25 and Figure 5-26 . The OFD for a normal operation is below the estimated threshold for most of the frames as shown in the plot. Method 2 took on average 3.95 s to process hopper region video dataset.

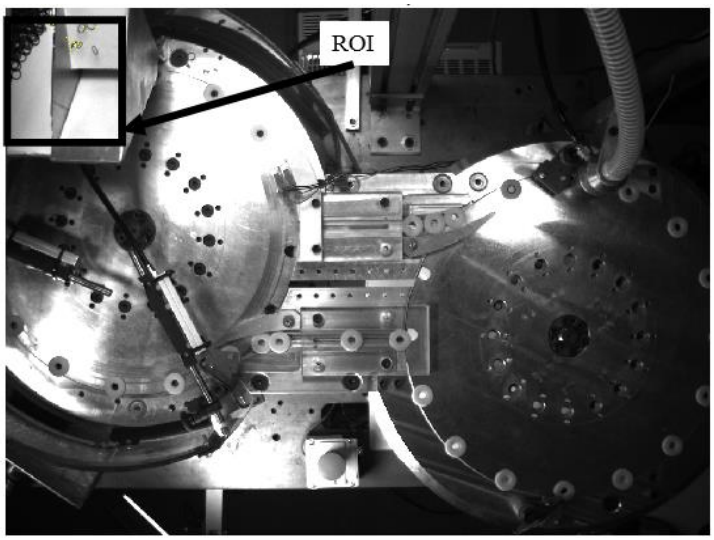


Figure 5-25 A hopper normal operation frame with ROI labelled (same as Figure 5-11, repeated for consistency)

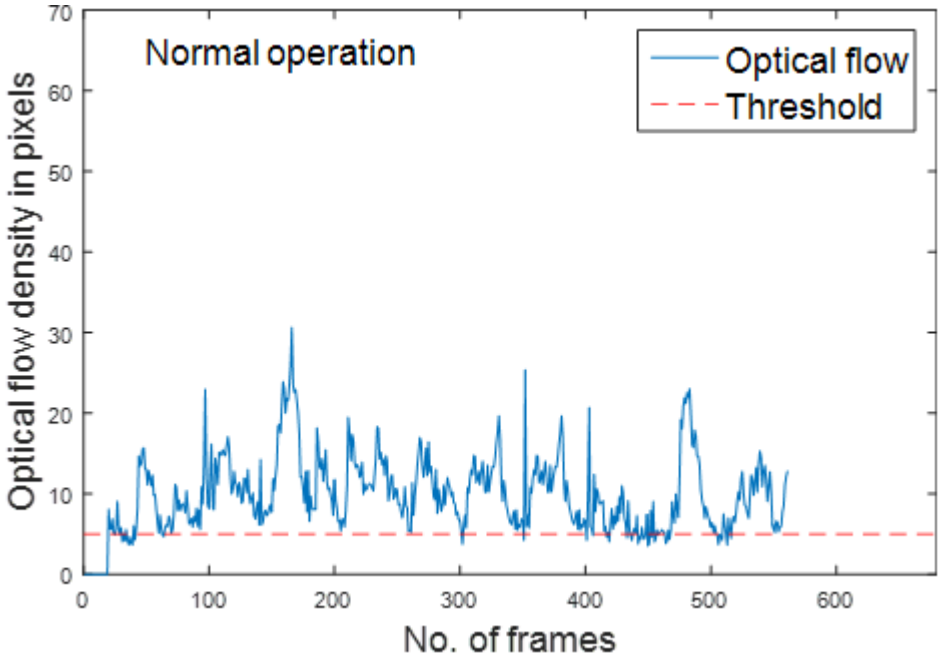


Figure 5-26 A hopper normal plot using Method 2

Hopper fault detection with the Method 2 is shown in Figure 5-27 and Figure 5-28. The fault is circled and the optical flow lines can be seen as short lines on the O-rings. The fault was introduced at frame no. 230 and it was detected at frame no. 278 with a delay of 48 frames. Method 2 applied to hopper video files resulted in 4 out of 10 correct classifications. Thus, the accuracy was 40 %.

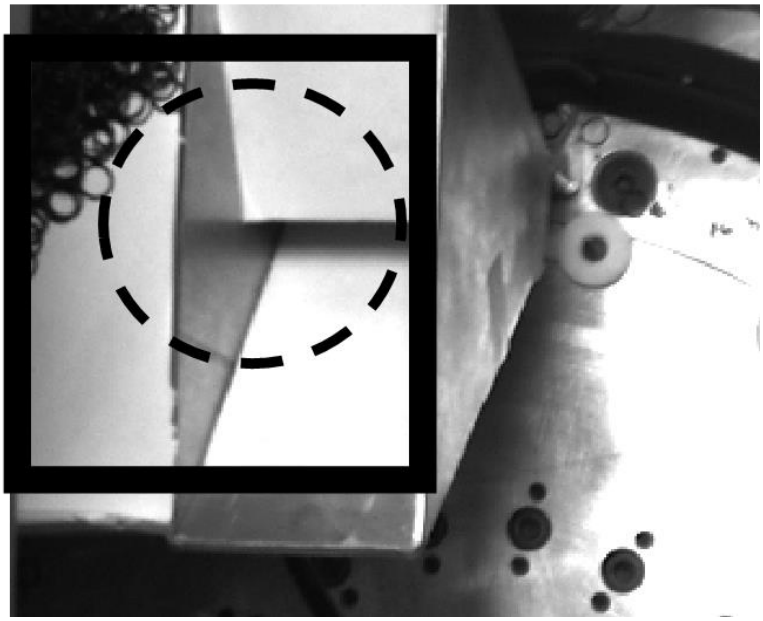


Figure 5-27 A hopper fault frame with fault circled within the ROI

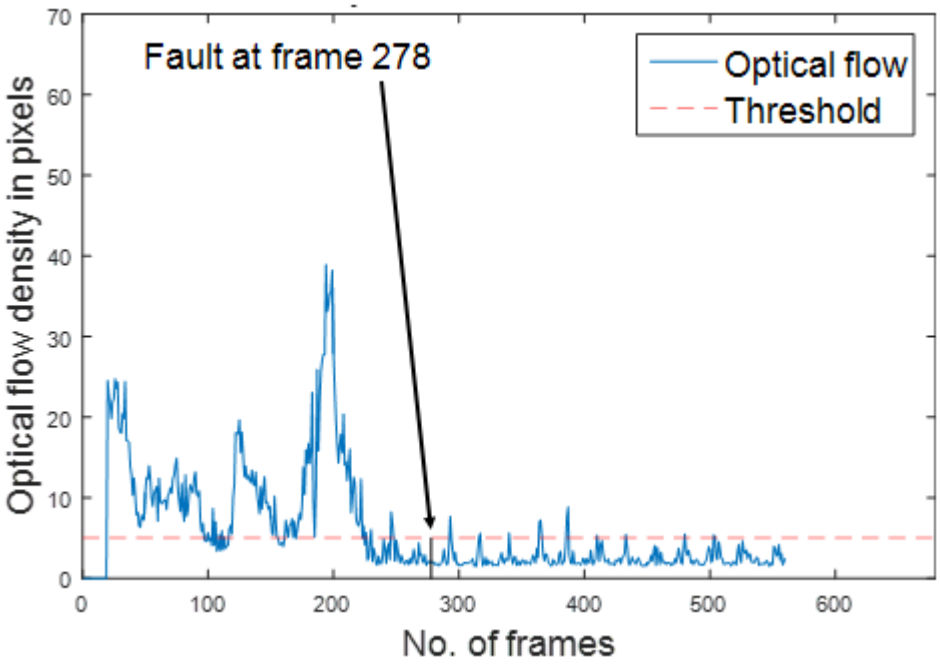


Figure 5-28 A hopper fault plot using Method 2 with fault detection frame identified

5.3 Method 3: Fault Detection with Foreground Running Average Area

Method 3 uses fast and low memory requirement background estimation technique for fault detection and classification. Unlike some background estimation methods such as a mixture of Gaussians or mean-shift, Method 3 used less computationally expensive way of background estimation. It used running average filter applied on the last 20 frames to obtain the background for the current frame. The method used preprocessing steps, prior to background estimation, such as spatial low pass Gaussian filter for noise removal (due to dark field lighting) and histogram equalization. It reads each image frame and implemented the above mentioned preprocessing steps. The method updated the background frame frequently, therefore, background image was robust to the motion and minor changes in the lighting conditions. The background estimation was followed by foreground detection and morphological operations to obtain the area of the moving objects. In this method, a fault in the machine was detected by measuring a running average area of a processed foreground from a ROI. The running average area for the current frame was calculated by averaging the foreground object area of the last 7 frames. The measured area was then compared against the threshold estimated from a set of training frames and if it exceeded the threshold for a few consecutive frames, a machine condition was classified as faulty, otherwise normal.

5.3.1 Transfer Track ROI Results with Method 3

The transfer track region has three operating conditions: a normal operation through the transfer track, transfer track 1 jam and transfer track 2 jam. The same video files as used in Method 1 are used as the examples for the fault detection with Method 3. For all examples, a sample frame image with the corresponding running average area (in pixels) vs frames plots are presented and discussed. Figure 5-29 and Figure 5-30 indicate an image with the transfer track region in the rectangle and a plot for a normal operation through the transfer track region. The running average area for the current frames is calculated by averaging the foreground area of the last 7 frames. The running average results in smooth output as shown in Figure 5-30. Moreover, the output is not zero even for the frames without the carriers due to the running average effect of the previous frames' results. The threshold value of the running average area was estimated with the training frames and it is plotted as dashed line in the plot. The dynamically estimated threshold value of the running average area for this normal operation is 800 pixels. The running average area from each frame was calculated and compared with the reference value. It can be seen in Figure 5-30 that the running average area for all frames is below the threshold throughout the operation. Hence, the operation was classified as a normal operation. Method 3 had correctly classified all transfer track region video files. Thus, the accuracy was 100 %. The Method 3 took on average 6.10s to process a 10s long transfer track video file.

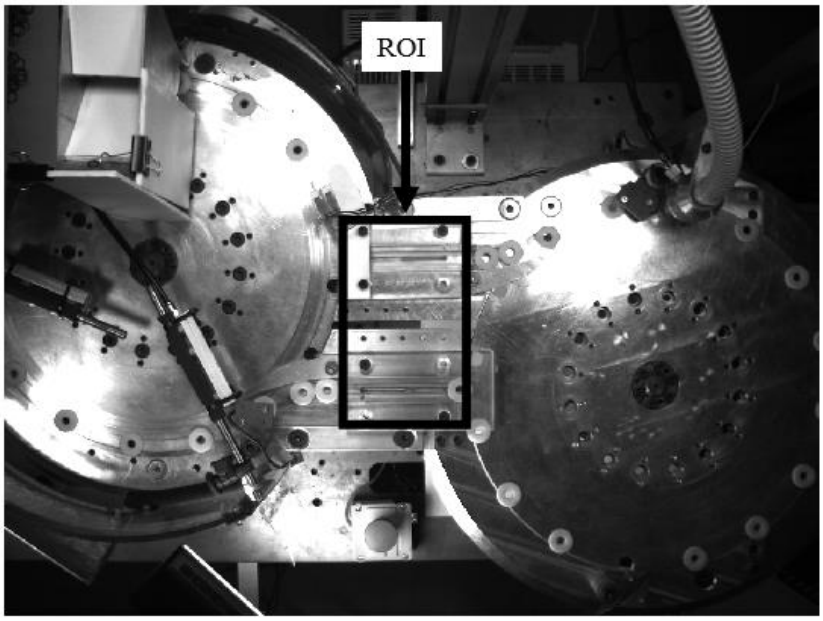


Figure 5-29 A transfer track normal operation frame with ROI labelled (same as Figure 5-1, repeated for consistency)

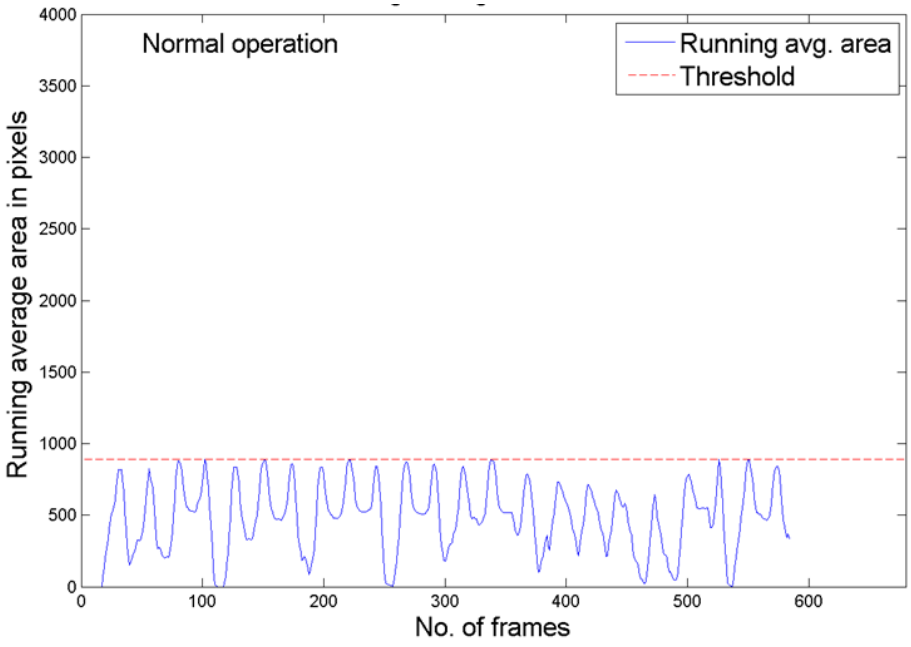


Figure 5-30 A transfer track normal operation plot using Method 3

Transfer track 1 jam detection with the Method 3 is shown in Figure 5-31and Figure 5-32. The fault is circled and the presence of carriers can be seen on the track. The fault was introduced at frame no. 300 and it was detected at frame no. 357 with a delay of 57 frames.

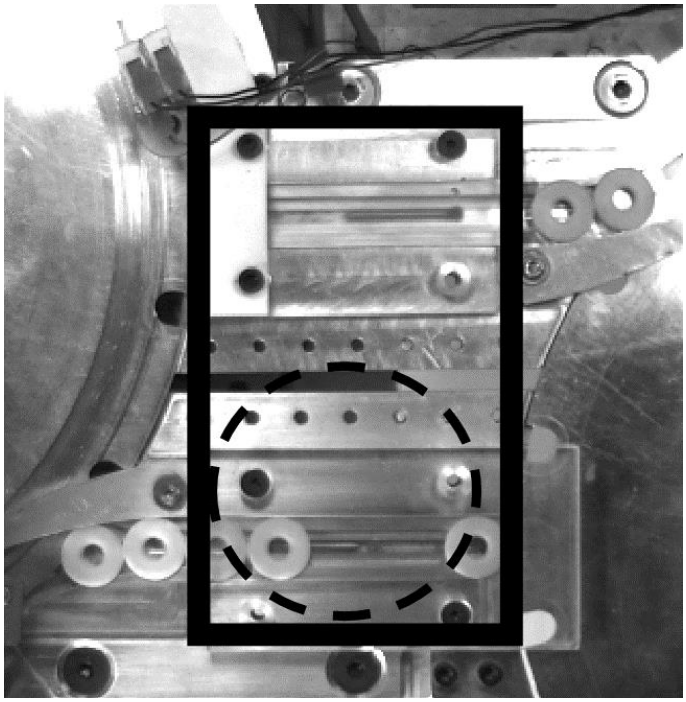


Figure 5-31 A transfer track 1 jam frame with fault circled within the ROI

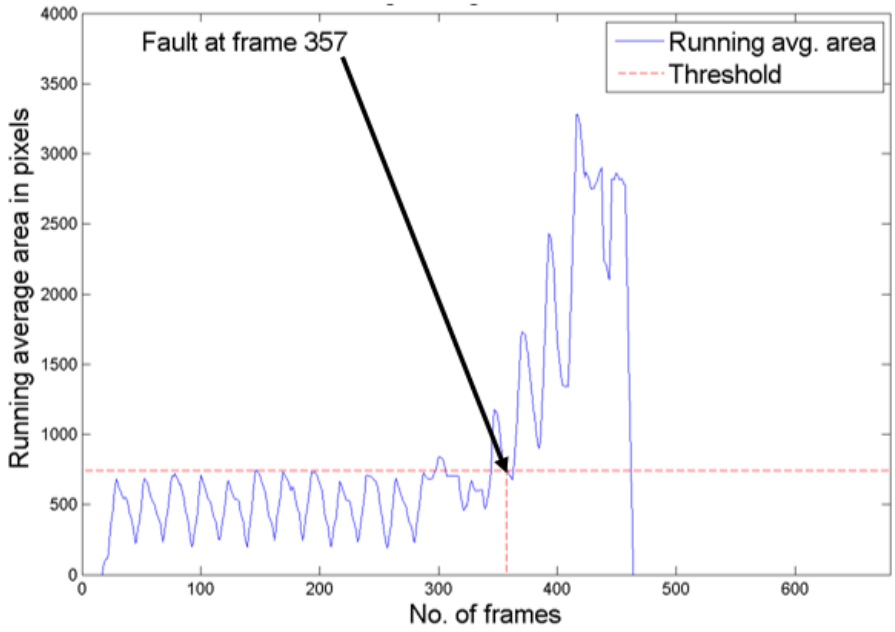


Figure 5-32 A transfer track 1 jam plot using Method 3 with fault detection frame identified

Transfer track 2 jam detection with Method 3 is shown in Figure 5-33 and Figure 5-34. The fault is circled and the presence of carriers can be seen on the track 2. The fault was introduced at frame no. 320 and it was detected by Method 2 at frame no. 342 with a delay of 22 frames.

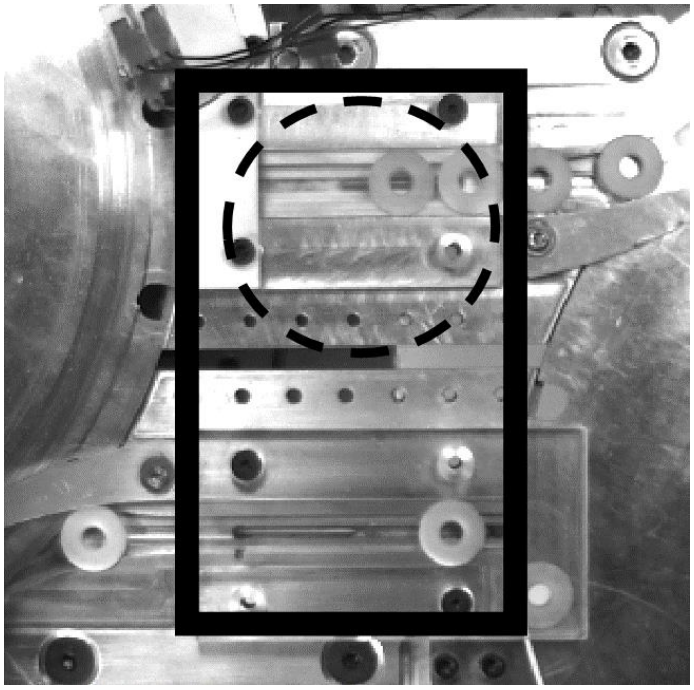


Figure 5-33 A transfer track 2 jam frame with fault circled within the ROI

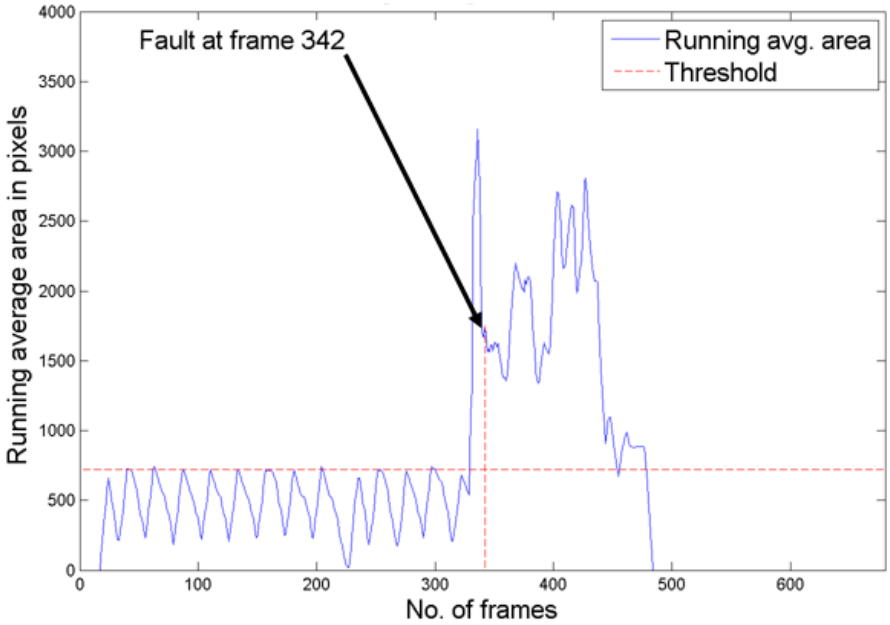


Figure 5-34 A transfer track 2 jam plot using Method 3 with fault detection frame identified

5.3.2 Air Knife ROI Results with Method 3

Air knife normal operation detection with Method 3 is shown in Figure 5-35 and Figure 5-36. The normal operation is indicated inside the box. The running average area for a normal operation is below the estimated threshold as shown in Figure 5-36. Method 3 took on average 3.15 s to process air knife region video file and resulted in 10 out of 10 correct classifications. Thus, the accuracy was 100 %.

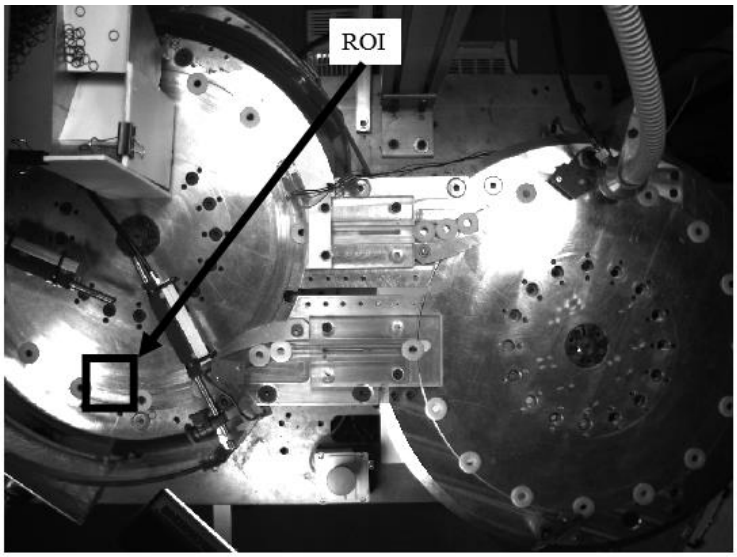


Figure 5-35 An air knife normal operation frame with ROI labelled (same as Figure 5-7, repeated for consistency)

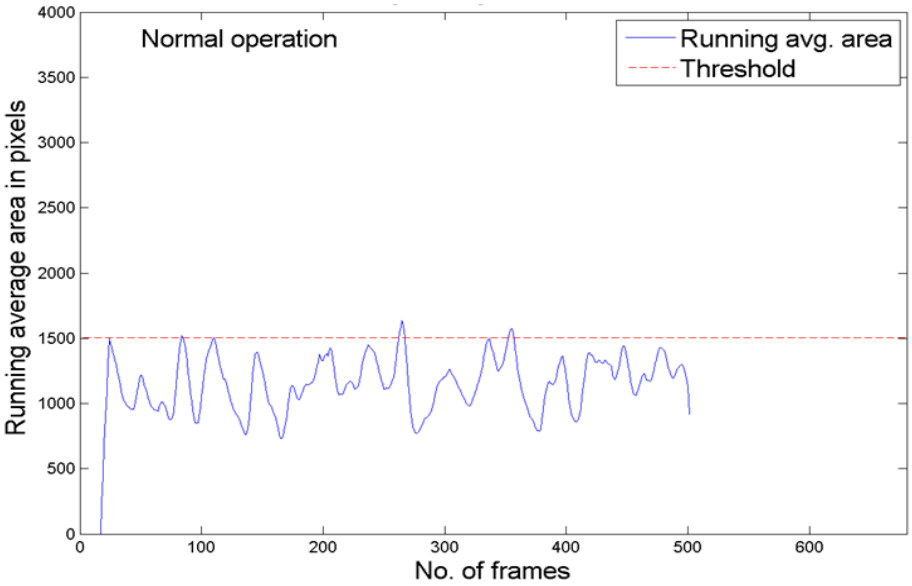


Figure 5-36 An air knife normal operation plot using Method 3

Air knife fault detection with Method 3 is shown in Figure 5-37 and Figure 5-38. The fault is circled. The fault was introduced at frame no. 290 and it was detected by Method 3 at frame no. 324 with a delay of 34 frames.

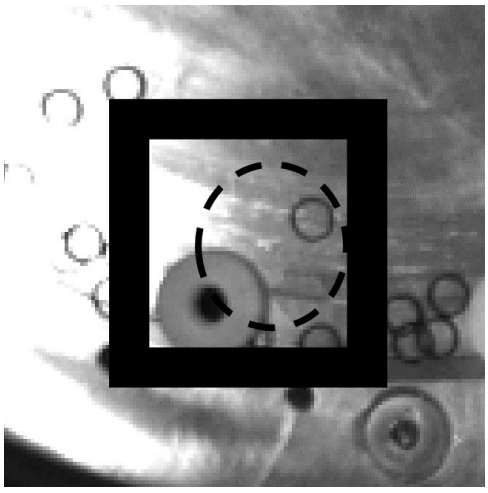


Figure 5-37 An air knife fault frame with fault circled within the ROI

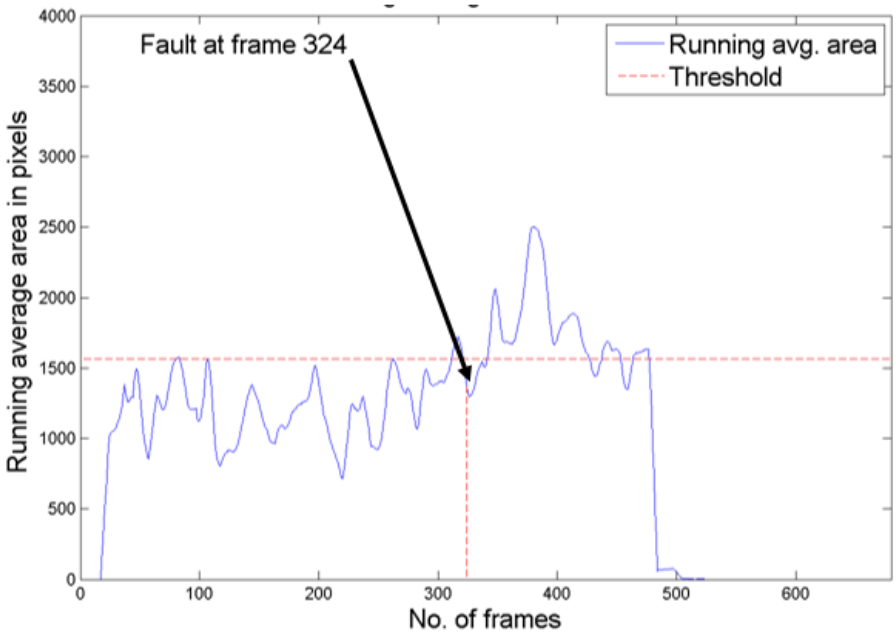


Figure 5-38 An air knife fault plot using Method 3 with fault detection frame identified

5.3.3 Hopper ROI Results with Method 3

Hopper normal operation detection with Method 3 is shown in Figure 5-39 and Figure 5-40 . The running average area for a normal operation is below the estimated threshold for all of the frames as shown in Figure 5-40. Method 3 took on average 3.95 s to process hopper region video files.

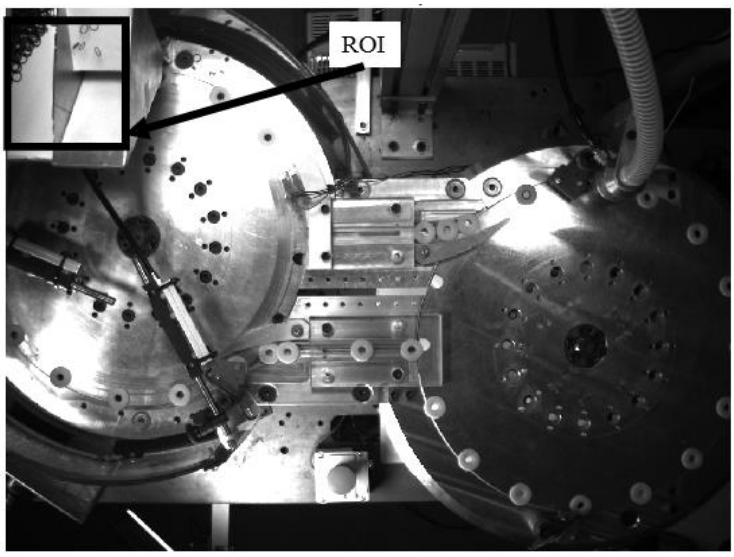


Figure 5-39 A hopper normal operation frame with ROI labelled (same as Figure 5-11, repeated for consistency)

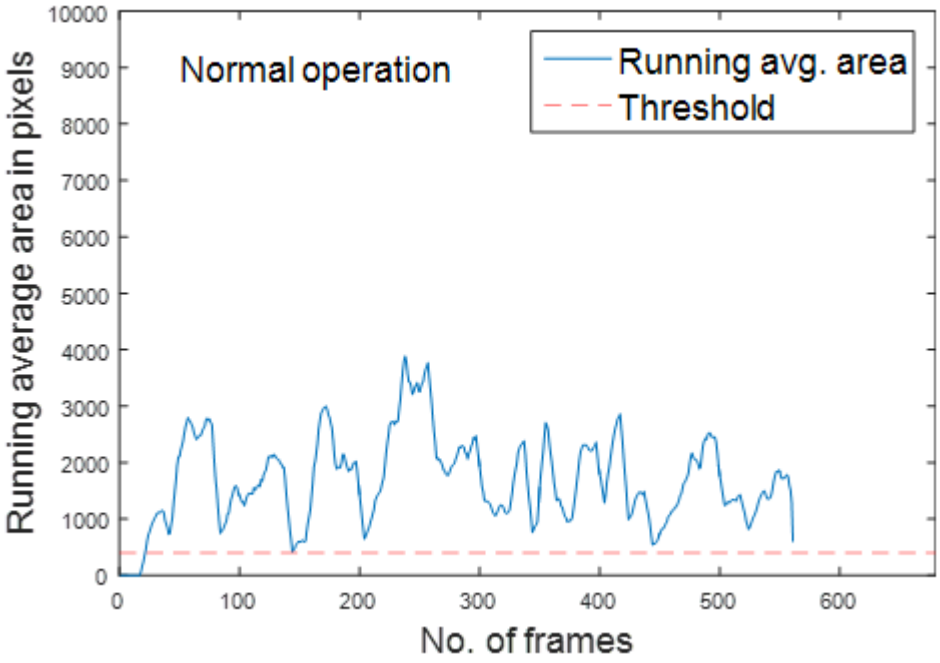


Figure 5-40 A hopper normal operation plot using Method 3

Hopper fault detection with Method 3 is shown in Figure 5-41 and Figure 5-42. The fault is circled. The fault was introduced at frame no. 230 and it was detected by Method 3 at frame no.302 with a delay of 72 frames. Method 3 applied to hopper video dataset resulted in 8 out of 10 correct classifications with one FN and one FP result. Thus, the accuracy was 80 %.

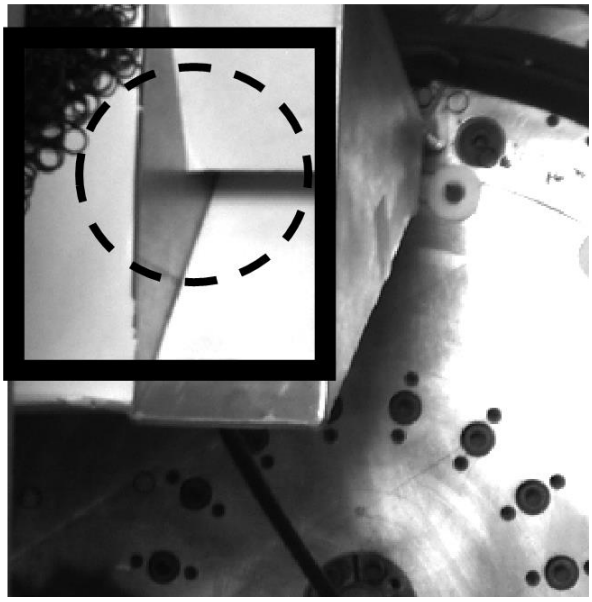


Figure 5-41 A hopper fault frame with fault circled within the ROI

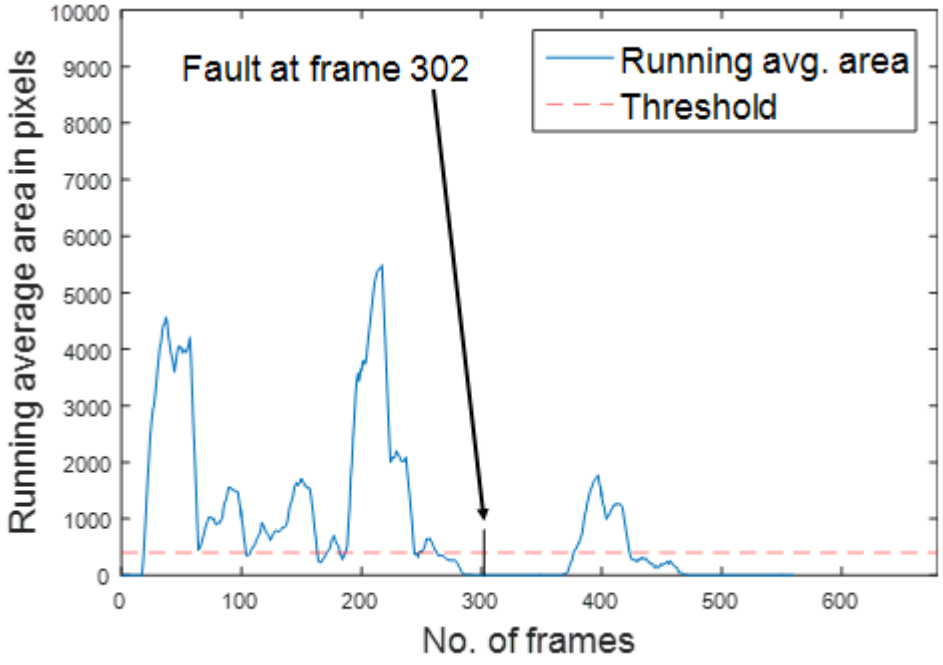


Figure 5-42 A hopper fault plot using Method 3 with fault detection frame identified

5.4 Accuracy and Detection time Results

The inspection results with the three methods are summarized in Table 5-2. The table lists for each method the average total frames in a video dataset, the number of correct detections for individual ROIs, average delay in fault detection and the total number of correct detections. Method 1 classified 35 out of 35 video files correctly. Method 2 classified 21 out of 35 video files correctly. Method 3 classified 33 out of 35 video files correctly. The individual results for each of the 35 video files and the corresponding dynamically calculated thresholds are given in Appendix D.

Table 5-2 Summary of accuracy and fault detection results

<table>
<tr><th rowspan="2">ROI</th><th rowspan="2">Average total frames</th><th colspan="3">Number of correct detections and total video files for the ROI</th><th colspan="3">Average delay in fault detection (frames) and time to process 10 s long video file (300 frames)</th></tr>
<tr><th>Method 1</th><th>Method 2</th><th>Method 3</th><th>Method 1</th><th>Method 2</th><th>Method 3</th></tr>
<tr><td>Transfer track ROI video dataset (15 video files)</td><td rowspan="3">577</td><td>15 out of 15</td><td>7 out of 15</td><td>15 out of 15</td><td rowspan="3">36 frames and 3.33 s</td><td rowspan="3">35 frames and 3.83 s</td><td rowspan="3">43 frames and 5.64 s</td></tr>
<tr><td>Air knife ROI video dataset (10 video files)</td><td>10 out of 10</td><td>10 out of 10</td><td>10 out of 10</td></tr>
<tr><td>Hopper ROI video dataset (10 video files)</td><td>10 out of 10</td><td>4 out of 10</td><td>8 out of 10</td></tr>
<tr><td colspan="2">Total correct detections</td><td>35 out of 35</td><td>21 out of 35</td><td>33 out of 35</td><td colspan="3"></td></tr>
</table>

5.5 Performance Comparison of three MVI methods

Fault detection and classification with three MVI methods was tested for seven operating conditions of the machine out of which three conditions were normal and four were faulty. All conditions were grouped into three ROIs: transfer track ROI, air knife ROI and hopper ROI. The ROIs were processed one at a time and the parameters of the three MVI methods were tuned for individual regions at first. Setting the parameters only for a specific ROI resulted in the perfect classification for that ROI. In the first stage of the algorithm development, tuning parameters for each method were adjusted to correctly classify only one fault condition at a time. Once, the method worked for one fault, the parameters were modified and the method was tuned to detect and classify the remaining faults with the same parameters. At this stage, the methods got separated in terms of their ability to classify the faults. As a result of the fixed parameters, the accuracy of classification dropped for Method 2 significantly and for Method 3 marginally.

In order to better compare the performance of the three methods, a novel Machine Vision Performance Index (MVPI) was developed. The MVPI is based on five measures of performance: accuracy of a method to classify the given machine condition, number of frames processed in a given time, speed of response, robustness against noise, and ease of implementation or tuning efforts. These five parameters are selected because a MVI systems needs time to process a video file and display frames, it requires a certain number of frames to see a fault, it can be sensitive to noise and it can be application specific. This means, when a MVI system developed for one application is applied to a different application, it needs to be retuned. The MVPI is the weighted sum of these parameters. It's hard to develop a single MVI method that could be the best in all five measures. For example, a MVI method involving complex steps can give the highest accuracy but it can be slow in terms of the processing speed of the algorithm. Hence, the MVPI serves as a more comprehensive way of comparing the performance of different MVI methods.

5.5.1 Overall Results with three MVI Methods

Each MVI method was tested with seven different machine conditions. For each machine condition, five video files were recorded. The overall inspection results with the three MVI methods are reported in terms of the following parameters.

- True Positive(TP) : Operation is normal and MVI method classifies it as normal
- True Negative(TN) : Operation is faulty and MVI method classifies it as faulty
- False Positive (FP) : Operation is faulty and MVI method classifies it as normal
- False Negative (FN) : Operation is normal and MVI method classifies it as faulty

A total of 35 video files were used to obtain these four parameters for each method. TP and FN results were obtained from 15 normal operation video files. TN and FP results were calculated from 20 faulty operation video files. Table 5-3 lists the four parameters for all three MVI methods.

The number of frames delay in fault detection by each method is very important parameter because the ability of a MVI method to prevent damage depends on the time taken by the method to detect and classify the fault. For the transfer track region, the results are shown in Figure 5-43. The solid line indicates the fault introduction frame numbers while the other three lines indicate fault detection frame numbers with each MVI method. A method with a line closest to the fault introduction line is considered as the fastest fault detection method. It can be seen that Method 1 fault detection line is nearest to the fault introduction line and hence it is the fastest method for transfer 1 jam detection. Similarly, transfer track 2 jam detection results are shown in Figure 5-44. Method 1 and Method 3 had performed equally well in transfer track 2 jam detection. Method 2 failed to detect 4 out of 5 transfer 2 jam operations.

Table 5-3 MVI methods classification results

MVI method	**Normal operation inspection results**		**Faulty operation inspection results**	
	TP	FN	TN	FP
Method 1	15	0	20	0
Method 2	7	8	14	6
Method 3	14	1	19	1

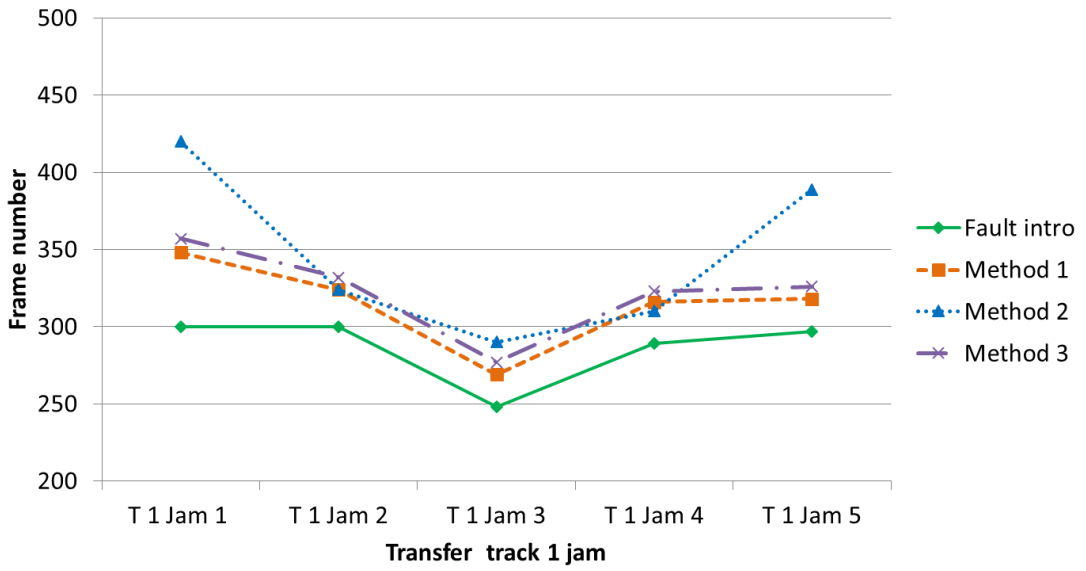


Figure 5-43 Transfer track 1 jam fault detection frame numbers

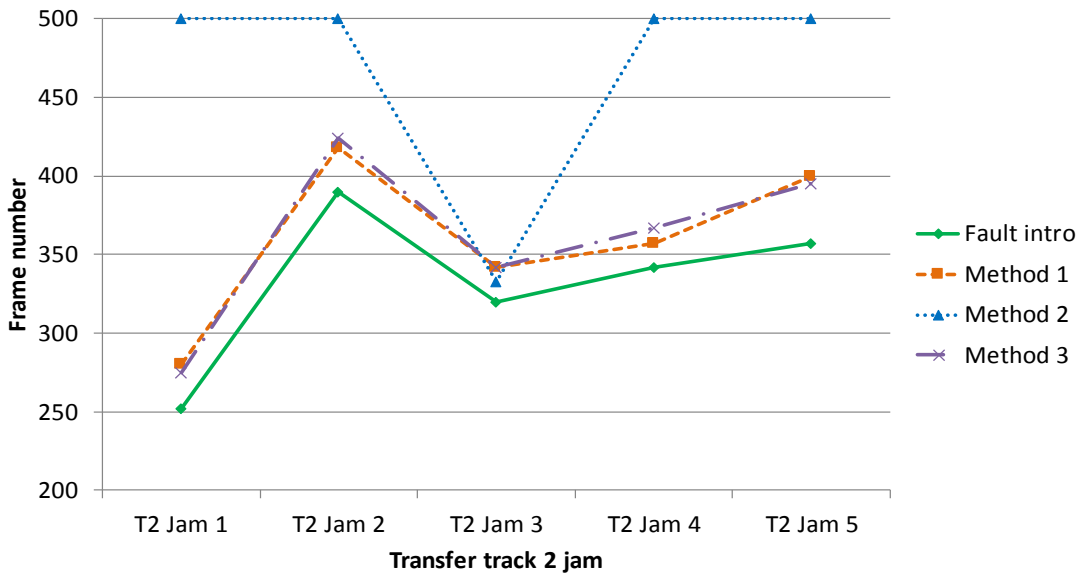


Figure 5-44 Transfer track 2 jam fault detection frame numbers

Air knife fault detection frame numbers are shown in Figure 5-45. All three methods had correctly classified the air knife faults but Method 2 performed best in terms of the speed of fault detection. Method 1 performed second best for air knife fault detection while Method 3 came in at third place. The air knife faults had small motions of O-rings on a moving background and optical flow is sensitive to small motions. Hence, it was expected that Method 2 would perform well for this fault because Method 2 used optical flow.

Hopper fault detection frame numbers are shown in Figure 5-46. Method 1 correctly classified all

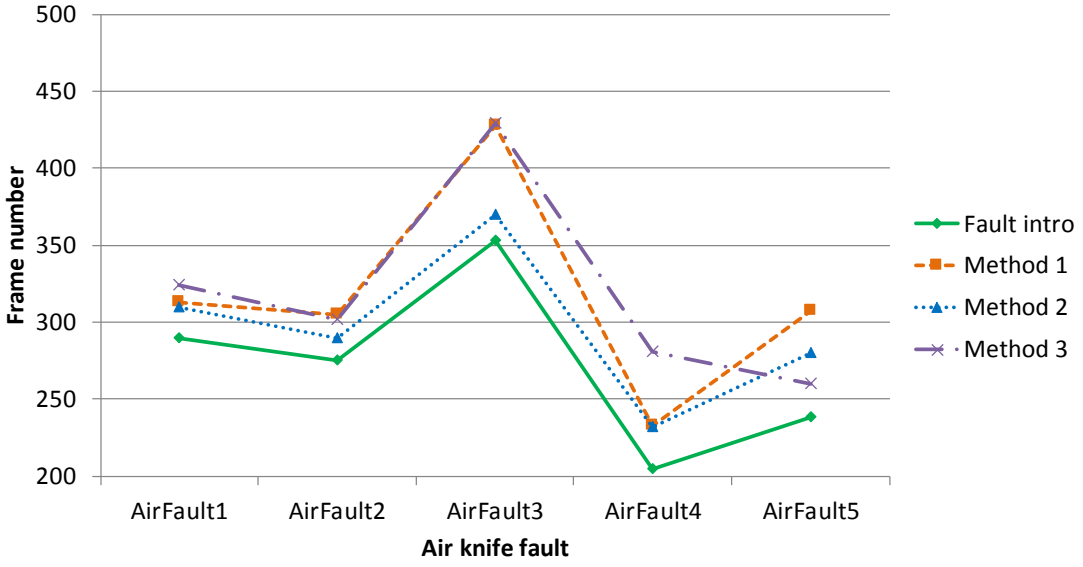


Figure 5-45 Air knife fault detection frame numbers

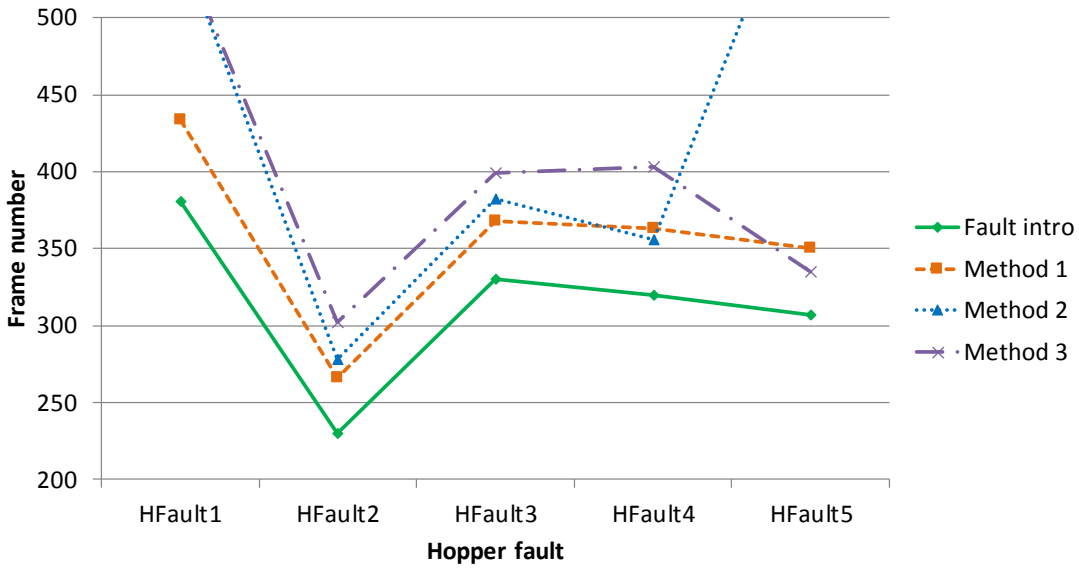


Figure 5-46 Hopper fault detection frame numbers

of the hopper fault video files. Method 2 failed for two video files. Method 3 failed for one video file. In terms of fault detection frame numbers, Method 1 had the lowest fault detection time and hence is considered better than the other methods for this measure. Method 2 classified three out of five results correctly and for these results it performed as well as Method 1. Method 3 classified only four out of five hopper fault video files correctly and it took more frames to detect and classify the fault compared to other methods. The reason for the delay in fault detection for Method 3 is partially due to the running average filter. The running average filter of size n provides a stable output for the current location only after processing the next (*n/2*) frames. For example, in Method 3 the running average filter of size 7 was used and that resulted in a stable output for any current frame after processing the next four frames.

5.5.2 Performance Parameters and MVPI

All three MVI methods worked perfectly in terms of accuracy when tuned to classify individual faults. Accuracy is one of the more important performance measures. But it is not the only performance measure. It is proposed that the overall performance should be calculated based on five performance measures: accuracy, processing time (measured in terms of number of frames processed in 10 s), speed of response to communicate a decision, robustness against noise, and ease of implementation or tuning efforts. These parameters are determined for all three methods and are then normalized and multiplied by weights to obtain the MVPI. Accuracy is assigned a higher weight compared to other parameters because it is of prime consideration.

Parameter p_1: Measure of accuracy in fault detection

Accuracy in fault detection and classification is the measure of the number of correct classifications. For this thesis, the accuracy is calculated based on the inspection results of 35 video files. Method 1 correctly classified all 35 video files into their correct categories. Hence, its accuracy is 100 %. Method 2 provided 21 correct classifications out of 35. Therefore, its accuracy is 60%. Method 2 misclassified 14 video files with 6 FPs and 8 FNs. Method 3 classified 33 out of 35 videos correctly with one FP and one FN result. Hence, the accuracy for Method 3 is 94 %. The accuracy results are illustrated in Figure 5-47.

Parameter p_2: Number of frames processed in 10 s

This parameter depends on the time taken by a method to process a single frame of the video file. A method with less time to process a single frame can process more number of frames in 10 s and it can handle high frame rates for video file acquisition. For this thesis, a frame rate of 30 fps was used but a higher fps may be required if the machine is required to operate at a higher speed. High fps provides better visualization in the case of a fault diagnosis because it can replay the motion with finer movement of the parts with respect to time. Hence, this parameter is an important consideration when selecting the fps of the camera relative to the machine speed. The number of frames results are illustrated in Figure 5-48. Method 1, 2 and 3 processed 900, 783 and 531 frames respectively.

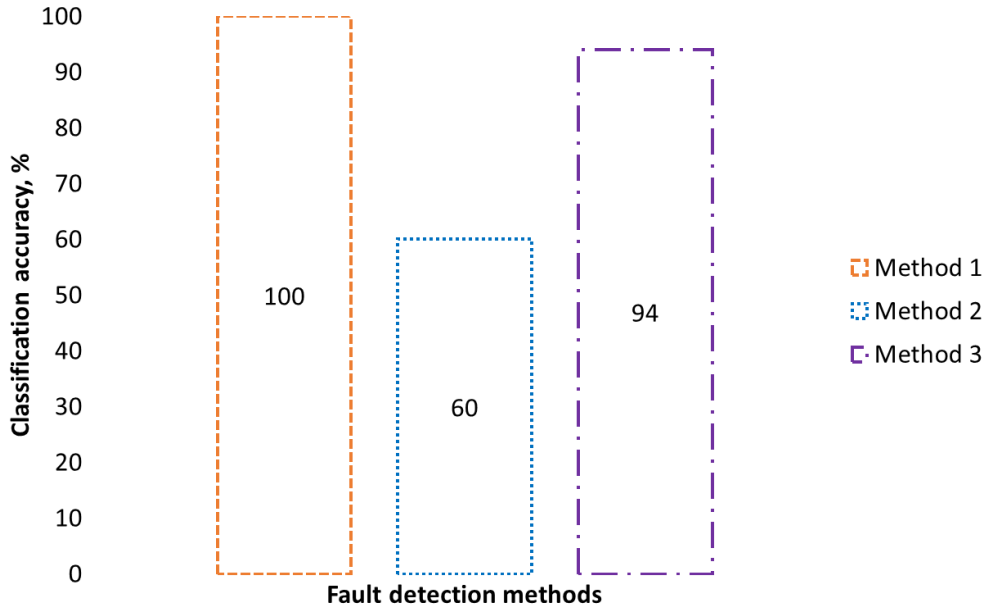


Figure 5-47 Performance parameter p_1: accuracy of classification

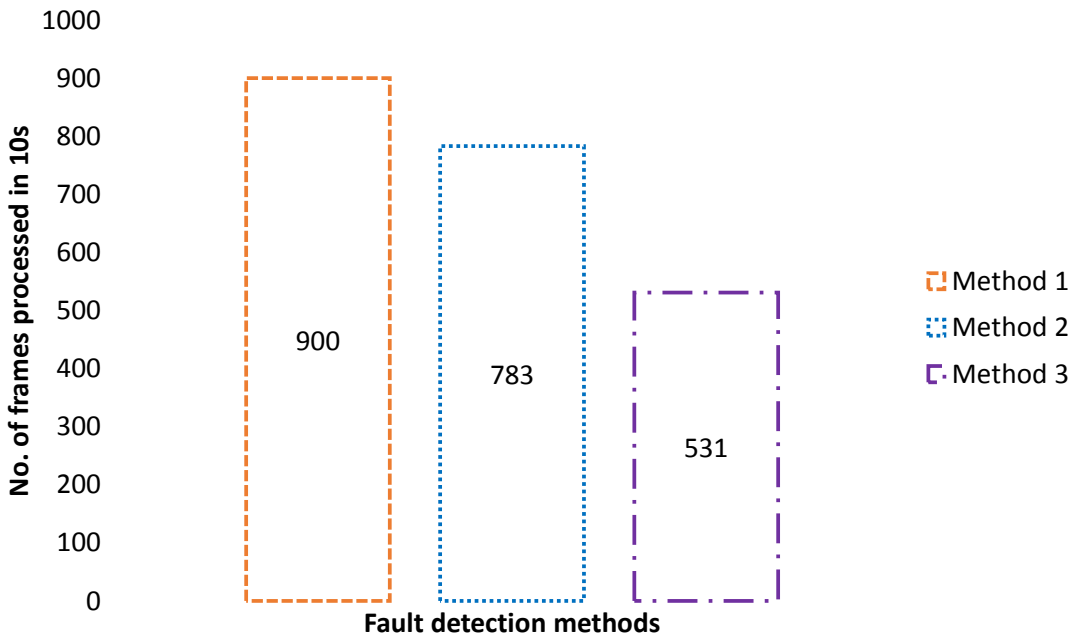


Figure 5-48 Performance parameter p_2: No. of frames processed in 10s

Parameter p_3: Measure of speed of response

This parameter is measured indirectly by measuring how fast a method detects the fault. Each MVI method needs a certain number of frames to see a fault after it was introduced. The number of frames required to detect the fault varied from a fault to fault and therefore an average value of the frames was used in the calculation. Method 1 required 36 frames to detect the fault. Methods 2 and 3 needed 35 and 43 frames, respectively. A MVI method that detects a fault sooner is in a better position to take necessary corrective action to prevent further damage. The parameter is scaled by dividing the number of frames required by each method with the frame rate of video file acquisition (i.e. 30 fps) and subtracting the time from 2 s. The time of 2 s was selected as the maximum available time to communicate the decision before serious damage could occur on the machine. The maximum time available was measured in practice by continue running the machine even after the fault. The sooner the MVI system communicates the decision, the lower will be the damage to the machine. Parameter p3 is the time available to communicate the decision. The comparison is shown in Figure 5-49. Method 2 proved to be the best with 0.84 s available compared to Method 1 (0.80 s) and Method 3 (0.57 s).

Parameter p_4: Measure of robustness against noise

Noise can affect the quality of images and consequently the response of a MVI system. Noise in images may result due to poor lighting, noise in the image sensor itself or noise in the transmission and reception of the image data. To study the effect of noise on the MVI methods, all video files were corrupted with Gaussian noise of zero mean and 0.01 variance. The unit of noise is pixel intensity. This is the source of noise in the images. All noisy video files were processed by the three MVI methods and accuracy results were recorded. It was observed that the accuracy for noisy video files with Method 1 was 54% (19 correct classifications out of 35), Method 2 and Method 3 was 14% (5 correct classifications out of 35). The parameter p_4 is the ratio of accuracy with noisy video dataset to accuracy without noisy video dataset. Method 1, 2 and 3 have the value of parameter p3 as 0.54, 0.24 and 0.14, respectively, as shown in Figure 5-50.

Parameter p_5: Measure of ease of implementation and tuning

MVI solutions are application specific and their tuned parameters depends on the type of camera, lens and lighting. Some parameters in an algorithm are fixed, while others are tunable. Hence, changing either the

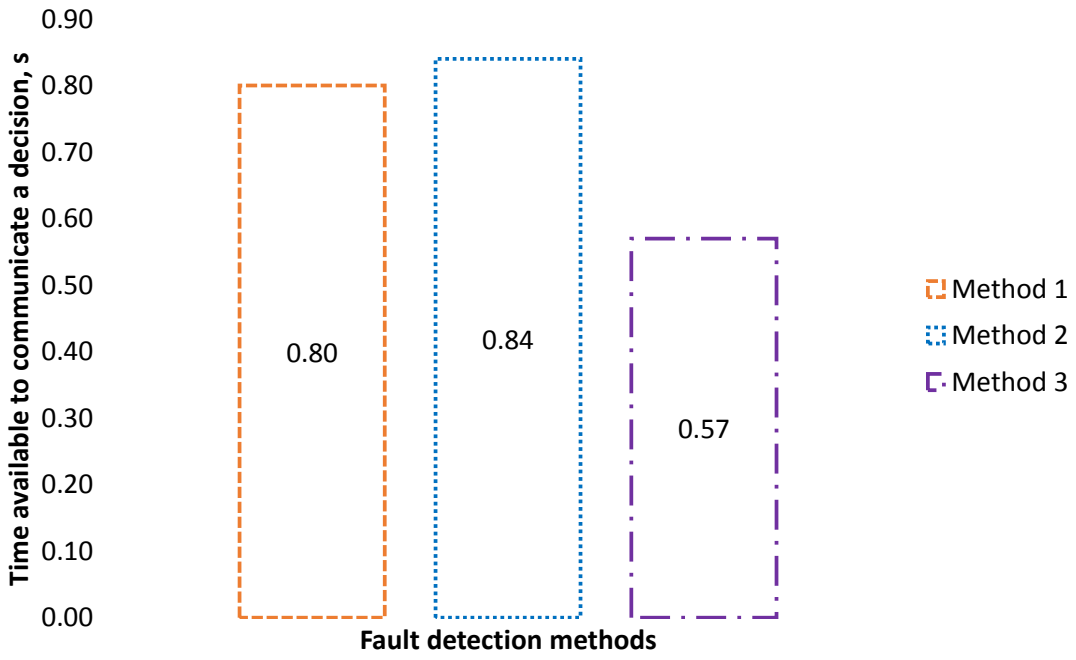


Figure 5-49 Performance parameter p_3: time available to communicate a decision

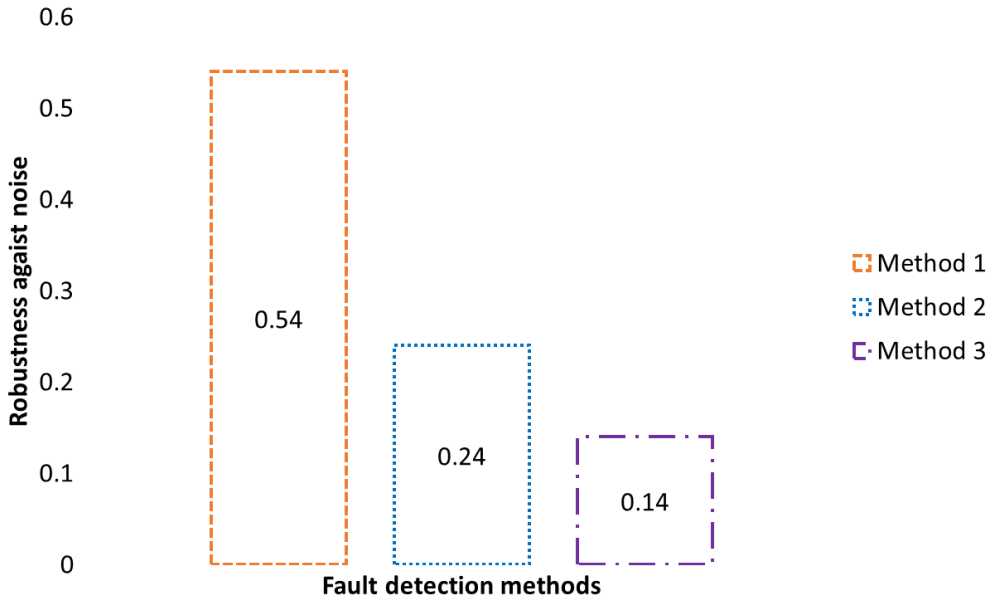


Figure 5-50 Performance parameter p_4: measure of robustness against noise

camera, lens and lighting for the same application, or changing to different application, requires retuning of the parameters. Method 1 has 5 tunable parameters. Method 2 has 4 tunable parameters. Method 3 has three tunable parameters. The parameters are listed in Table 5-4. The iterative tuning procedure and tuned numerical values are given in Appendix E. The ease of implementation and tuning is inversely proportional to the number of tunable parameters. The parameter p_5 is calculated by multiplying the inverse of the number of tunable parameters with the scale of 2 for each MVI method. Hence, for Method 1 it is 0.4. For Method 2 it is 0.5. For Method 3 it is 0.7. Figure 5-51 illustrates these results.

The MVPI is the weighted sum of these five performance parameters and is calculated using Equation (5.1).

Table 5-4 Tunable parameters

Sn.	Method 1	Method 2	Method 3
1	No. of frames used for background estimation	Optical flow components type	Running average filter size for background estimation
2	Initial variance for GMMs	Subsampling size for optical flow field	Filter size for running average area calculation
3	Minimum blob area to be detected	Scaling of flow fields	No. of frames used to estimate a threshold
4	Maximum blob area	No. of frames used to estimate a threshold	
5	No. of frames used to estimate a threshold		

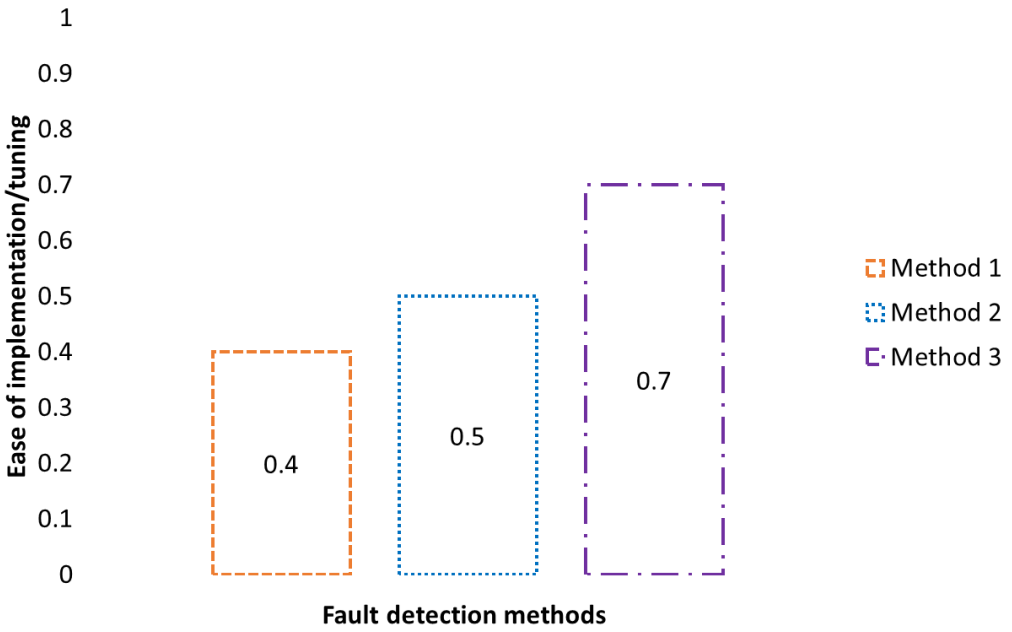


Figure 5-51 Performance parameter p_5: ease of implementation/tuning

$$P = \frac{W_a \times p_1 + W_b \times p_2 + W_b \times p_3 + W_b \times p_4 + W_b \times p_5}{(W_a + 4 \times W_b)} \times 100 \qquad (5.1)$$

where, Ws (W_a =100 and W_b =50) are the weights which determined the relative importance of the performance measure. W_a is double that of W_b as accuracy is more important than the other parameters. The weight of 2 was selected for illustrative purposes. As long as W_a was greater than W_b, it was found that the relative ranking of the three methods did not change.

Using Equation (5.1) and the values from Table 5-5, the MVPI is calculated for each method. The MVPI for Method 1 is 77, Method 2 is 58 and Method 3 is 64. These results are illustrated in Figure 5-52. The uncertainty analysis for MVPI calculation is given in Appendix F. Method 1 achieved the highest MVPI. Method 2 had the lowest MVPI. The MVPI gives the relative ranking of the methods based on the five assigned performance measures.

Table 5-5 Performance parameters for MVPI

Sn.	Performance parameters	Method 1	Method 2	Method 3
1	p1: Classification accuracy	1.00	0.60	0.94
2	p2: No. of frames processed in 10s	0.90	0.78	0.53
3	p3: Speed of response	0.80	0.84	0.57
4	p4: Robustness against noise	0.54	0.24	0.14
5	p5: Ease of implementation/tuning	0.40	0.50	0.70

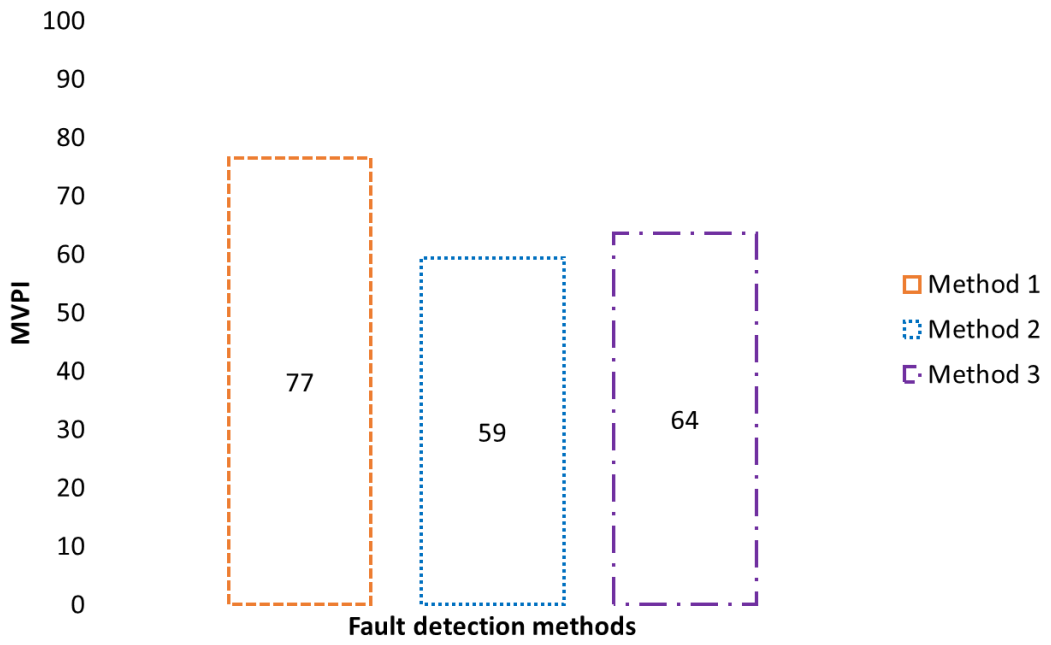


Figure 5-52 MVPI for three MVI fault detection methods

5.6 Summary

The results for three MVI methods for fault detection and classification in an automated assembly machine were presented in this chapter. The faults were grouped into three ROIs: transfer track ROI, air knife ROI and hopper ROI. Each ROI covered both normal and faulty operations of the assembly machine. The transfer track ROI covered normal operation through the transfer tracks, transfer 1 jam and transfer 2 jam. The air knife region covered normal operation through the air knife ROI and air knife fault. The hopper ROI covered hopper normal operation and hopper fault. One sample image and the full plot of the features vs frames were presented for all three methods. The overall performance in terms of the classification accuracy was reported.

It was observed that Method 1 successfully classified all 35 operating conditions. Method 3 classified 33 out of 35 operating conditions correctly with in one FP and one FN result. Method 2 classified 21 out of 35 operating conditions correctly with 8 FN and 6 FP results. The performance comparison of the three MVI methods was calculated based on five performance measures: accuracy, processing time, speed of response, robustness against noise, and ease of implementation. These parameters were weighted and the overall performance was reported in terms of the MVPI. The overall performance of three MVI methods based on the MVPI was reported. This ranked Method 1 as first, followed by Method 3 as second and Method 2 as third.

Chapter 6

Conclusions and Recommendations

Fault detection and classification in an automated assembly machine using a MVI system was the subject of this thesis. A high speed automated assembly machine was used as a test apparatus. The machine assembled small diameter circular O-rings onto continuously moving plastic carriers. Three normal and four faulty operating conditions of the machine were identified and five videos were recorded for each of the conditions. The ROIs were defined on the machine surface in such a way that, it covered all the potential faults on the machine. A PLC with HMI was used to introduce the controlled faults on the machine to record the video data for faulty operating conditions. Once all data were acquired, the video files were processed offline using MATLAB's image processing and computer vision toolboxes.

The machine used a pneumatic system for vacuum suction and transfer of the assemblies and empty carriers. The four faults were transfer track 1 jam and transfer track 2 jam, air knife fault and the hopper fault. These faults were grouped into three ROIs, namely transfer track region, air knife region and the hopper region, with respect to the FOV of the camera. A PLC with HMI was used to introduce controlled faults to acquire the video data for faulty operations. A total of 35 videos were recorded for the three ROIs. The transfer track region had 15 video files: 5 video files for a normal operation through the transfer track region, 5 video files for transfer track 1 jam and 5 video files for transfer track 2 jam. The air knife region had 10 video files: 5 video files for a normal operation through the air knife region and 5 video files for the air knife fault. The hopper region had 10 video files: 5 video files for a normal operation through the hopper region and 5 video files for the hopper fault. For the faulty operation videos, the machine was run under the normal operating condition for up to 200 frames and then the faults were introduced by blocking the air supply to the appropriate solenoid valves. All video files were processed by three MVI algorithms to detect and classify the machine's operations into their correct categories.

The three MVI methods were: (1) fault detection using GMMs and blob analysis, (2) fault detection using optical flow estimation and (3) fault detection using foreground running average area were tested with all video files for detection and classification of the machine's operating condition. Method 1 used GMMs for foreground estimation and it detected only the moving objects (foreground) from the frames. The area of the foreground was calculated using the blob analysis approach. The area was used as a feature to classify an operating condition into a normal or a faulty operation. Method 2 was based on optical flow for motion estimation from a video. The method used Horn-Schunck approach to solve optical flow

equations and measured the amount of the motion using the OFD. The OFD feature was utilized to detect and classify an operating condition of the machine into its correct category. Method 3 was based on a background estimation using the running average and subsequent morphological image processing steps. The method calculated the running average area of the moving objects and used that area as a feature for fault detection and classification on the O-ring assembly machine.

6.1 Effectiveness of MVI system Hardware

The main hardware and software elements of a MVI system are camera with lens, lighting and image processing software. There were many options available for selecting the MVI camera but a USB 3.0 digital camera was selected because of its low CPU load, low power consumption, high bandwidth and low cost. Once a camera is selected, the design of the light is critical. Light selection is dependent upon the part size, part color, surface features, geometry, inspection environment, and the system needs. The selection of proper lighting for the MVI system required the consideration of factors such as light source, lighting techniques, light directions, light colors and polarization. A rigorous lighting analysis was done based on the above five factors and a decision was made to use the LED panel lights as the lighting source for the project. The LED panel light was effective for the research project because of its flexibility, long life, stability, less heat generation and lower cost. The machine had shinny surfaces in the form of transfer wheels that resulted in specular reflection due to the lights. To avoid these bright reflections and to illuminate only the required FOV effectively, the LED lights were positioned as the dark field lights (calibration procedure is given in Appendix G) and the circular polarizer was mounted on the camera.

Overall the camera-lens and lighting were effective in monitoring the machine for fault detection and classification. They were relatively inexpensive and easy to implement. MATLAB's image acquisition toolbox was used for data acquisition while the image processing and computer vision toolboxes were used for video processing. MVI algorithm implementation and testing was relatively easy with MATLAB due to its built-in libraries and functions.

6.2 Practicality of MVI for Fault Detection and Classification

Three MVI methods were developed and tested. The video files were recorded for three normal and four faulty operating conditions. Each method was initially tuned to detect and classify individual faults. When tuned specifically for an individual fault, all three methods worked perfectly in classifying the faults. The faults were introduced as per the sequence: 1) transfer track ROI faults 2) air knife ROI faults and 3) hopper ROI faults. For the second stage of training and testing, the parameters were fixed for all MVI methods and they were tested against the full set of faults. A significant difference in the performance was noticed using

the fixed parameter approach for Method 2. The method was able to classify only 21 out of 35 video files correctly. Method 1 and Method 3 performed much better than Method 2.

All MVI methods learned normal behavior of the machine from the initial frames of each video during training and testing. They did not require separate training and testing data sets. The threshold value of the features was dynamic in the nature. i.e. the threshold value was not fixed and it was estimated from a set of initial frames from a single video file. Hence, the threshold value varied from one video file to another. The variation in threshold values for all 35 video files is given in Appendix D. The MVI methods were robust against changes in the light intensity within a small range. Significant changes in the light intensity either made the parts non-visible or too bright to be detected by the methods. MVI systems are sensitive to noise in the images. Therefore, the MVI methods were tested with noisy data set. Gaussian noise with a zero mean and 0.01 variance was added to all video files and the classification accuracy was calculated by each method. The addition of noise degraded the performance of all methods. Hence, for a MVI system to be practical, lighting conditions should not change by a considerable amount and the noise should be filtered.

The performance of the MVI methods was reported in the form of the MVPI based on five performance measures: accuracy of a method to classify the given machine condition, number of frames processed in a given time, fault detection speed after fault introduction, sensitivity to noise, and ease of implementation/tuning. The MVPI was the weighted sum of these parameters. For a MVI system to be practical, the first priority should be high accuracy. The other important characteristics are the time taken to process images and the speed of fault detection. For the MVI to work online in real time, the amount of time a method takes to process a single frame is a key parameter, as that decides the image acquisition rate. The USB 3.0 camera acquired images at a rate of 30 fps. That means a new frame was acquired at every 33.33 ms. For a MVI to be practical, it should not take more than 33.33 ms to process a frame; otherwise it would miss certain frames. Method 1, Method 2 and Method 3 took 11.11 ms, 12.82 ms and 18.76 ms, respectively. Therefore, they were practical for the fault detection on the O-ring assembly machine. The speed of fault detection was important because if the machine was not stopped in 2 s from the fault introduction, it could be damaged. Thus, the MVI should not only detect but it should also communicate the decision to the controller within 2 s to prevent the damage. The faster the MVI system communicates its decision, the lower will be the risk of damage. Method 1, Method 2 and Method 3 detected faults in 1.25 s, 1.43s and 1.16 s, respectively. Therefore, they could be used in real time.

Method 1, the GMMs based method, was found to be the most practical for fault detection and classification on the O-ring assembly machine. The method was able to classify all video files correctly and it was practical both in terms of the processing time and the speed of fault detection. The method was also

relatively robust against noise and was less sensitive to lighting variations. It had the highest overall MVPI with respect to the other two methods. One drawback of this method was that it had five tunable parameters. For the given application, these parameters were tuned and fixed but for any changes in the operation sequence or for a new application, these parameters would require retuning. The amount of time and effort required to tune a method depends on the number of tunable parameters. Therefore, retuning of Method 1 would take more time compared to the other two methods. The MVI methods could be used for similar vision inspection problems such as jam or missing part detection in automated bottle filling plant, steel picking process, PCB inspection and rotary feeders. However, MVI solutions are application specific and changing the application requires retuning both in terms of hardware (camera-lens and lighting) and software (retuning the parameters in the code).

6.3 Contributions

The first method was based on GMMs and blob analysis for fault detection and classification. GMMs technique is a statistical method of clustering a large number of overlapping data. In computer vision GMMs were originally developed for moving object detection from a video (Stauffer & Grimson, 1999). The GMMs method have also been used for vehicle detection and crowd monitoring. For the O-ring machine application, GMMs were used to estimate the foreground objects (O-rings and carriers). Parameters such as learning rate, number of training frames, number of Gaussian models in the mixture model and initial mixture model variance were tuned for the O-ring machine application. Post-processing steps such as: blob analysis of the estimated foreground to detect blob area feature and corresponding bounding box were implemented. The feature was tracked through the video frames and it was used to classify the machine's condition as normal or faulty.

The second method used an optical flow approach for fault detection and classification. The optical flow was originally developed for motion estimation from a video (Horn & Schunck, 1981). It has been used for applications such as vehicle tracking, video stabilization and video compression. For the O-ring machine application, the Optical Flow Density (OFD) feature was computed based on the motion estimation using optical flow. The OFD was tracked through the video frames and it was used to classify the machine's condition as normal or faulty. The use of optical flow for fault detection and classification on a machine is novel.

The third method was based on basic image processing techniques and included image averaging and morphological operations. Image averaging was originally developed for noise reduction in a video. For the O-ring machine application, an image averaging technique was used to estimate a background image because moving objects are appear as noise for this technique. Image processing steps of spatial filtering

and enhancement using histogram equalization were added prior to background estimation. Then, the background subtraction and object detection using morphological operations was implemented. The running average area of the detected objects was used as a feature. For the O-ring machine, the running average area feature was tracked through the video frames and it was used to classify the machine's condition as normal or faulty.

Computer vision techniques such as: GMMs, optical flow and image averaging could have not been used in their original form for the O-ring machine application. They were used as one of the steps for the three fault detection methods developed in this thesis. Additional steps of pre-processing, post-processing and feature extraction were necessary in order to use the original computer vision techniques for fault detection and classification.

The experimental results showed that all three MVI methods were capable of correctly classifying a given fault if their parameters were tuned for that specific fault. Under these circumstance, reporting performance based on accuracy only did not identify which method was better. That is to say, they were all highly accurate. Therefore, the performance of the MVI methods was evaluated using a novel performance index called Machine Vision Performance Index (MVPI). The MVPI was developed based on five measures of performance: accuracy, processing time, speed of response, robustness against noise, and ease of implementation. The MVPI was calculated by the weighted average of the performance parameters. The conclusions of the evaluation using the MVPI are: (a) single method was not the best in all performance criteria, (b) the accuracy was significantly reduced for all MVI methods with noisy input data and (c) the GMMs-based method had the highest overall performance result.

To summarize, the major contributions of this research were:

i. MVI system with a single camera was designed to detect and classify a variety of faults on an automated assembly machine using computer vision techniques. Three MVI methods based on GMMs, optical flow and foreground running average area were developed. The methods are adaptive in the sense that they learn the normal behaviour of the machine from a set of initial frames in the video. They do not require separate training and testing data sets.

ii. The MVPI for performance evaluation and relative performance comparison amongst multiple MVI methods was developed and applied to the task of fault detection and classification.

iii. The inherent value of a camera-based system compared to traditional sensor-based systems was explained. For example, during a video playback for a transfer track jam fault, it was discovered that one of the reasons for the jam was a loose O-ring that obstructed the flow of assemblies through the narrow path. Another reason of transfer track jam was primary and secondary wheel binding.

Other contributions of this research were:

i. Development of the MVI system using a USB 3.0 camera and four LED lights is not only beneficial for the O-ring assembly machine inspection but it can also be used for other applications such as rotary bowl feeder inspection, jam detection on transfer lines and missing component detection in assemblies.

ii. The GUI based MVI algorithms and MVPI can be used by other researchers in the field of machine vision to evaluate their own MVI applications. However, the MVI algorithms will need to be tuned for these applications. The tuning will include adjusting the number of training frames, the initial variance for GMMs, the density of optical flow, threshold estimation and the running average filter size. In addition to algorithm tuning, hardware tuning such as setting the camera focus, working distance, aperture and light intensity will also be necessary for any new application.

6.4 Limitations of MVI Methods

The developed MVI methods have the following limitations.

- The methods are able to detect and classify only faults with visual characteristics.
- The methods need faults to build up for certain frames after their introduction.
- The methods process only one ROI at a time. As a result, they are unable to monitor and detect simultaneous occurrences of multiple faults.
- The methods learn normal operation sequence from a set of initial frame. Hence, the machine should run fault free during initial frames of video acquisition.
- The accuracy of inspection has dropped significantly with noisy data.
- MVI-based methods are sensitive to variations in lighting. They require consistency in light during training and testing phases.
- If the system is to be used to monitor a machine that runs continuously for a long period of time. The threshold values should be updated at a regular interval of time.

6.5 Future work

The MVI methods used in this study were able to detect and classify seven different operating conditions of the machine. All methods used a single USB 3.0 camera that was mounted above the machine. The MVI methods processed three ROIs separately. That means, when a method was processing for faults in the transfer track region, it did not look for the faults in the air knife region. This could be the subject of the

future work, to expand the system such that it can see simultaneously look for multiple faults on the machine.

The MVI methods were able to detect and classify only known faults because the parameters were tuned for those faults. An unknown fault will have a different characteristic and it will require retuning the parameters of the MVI methods. Therefore, the limitation of the existing methods was their inability to handle unknown faults. Future work should look at what is required for the system to handle unknown faults. Deep Neural Networks (DNNs) could be used to detect unknown faults with their complex structure composed of multiple processing layers and multiple non-linear transformations (Krizhevsky et al., 2012).

References

AIA Vision Online. (Accessed: March 14, 2016). *Vision Education:Computer Vision vs. Machine Vision*. Retrieved from http://www.visiononline.org/vision-resources-details.cfm/vision-resources/Computer-Vision-vs-Machine-Vision/content_id/4585.

Batchelor, B. G. (2012). *Machine Vision Handbook.* Chapters 10-11, London: Springer.

Benezeth, Y., Jodoin, P., Saligrama, V., & Rosenberger, C. (2009). Abnormal Events Detection Based on Spatio-Temporal Co-Occurences. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2458-2465.

Bhuvanesh, A., & Ratnam, M. (2007). Automatic Detection of Stamping Defects in Leadframes Using Machine Vision: Overcoming Translational and Rotational Misalignment. *The International Journal of Advanced Manufacturing Technology, 32*(11), 1201-1210.

Blob Analysis. (Accessed: November 24, 2015). Retrieved from http://what-when-how.com: http://what-when-how.com/introduction-to-video-and-image-processing/blob-analysis-introduction-to-video-and-image-processing-part-1/

Boothroyd, G. (2005). *Assembly Automation and Product Design* (2nd ed., Vol. 66). Florence: Marcel Dekker. New York.

Braggins, D. (2000). Illumination for Machine Vision. *Sensor Review, 20*(1), 20-23.

Chauhan, V., Fernando, H., and Surgenor, B. (2014). Effect of Illumination Techniques on Machine Vision Inspection for Automated Assembly Machines. *Proceedings of The Canadian Society for Mechanical Engineering (CSME) International Congress, Toronto,* 1-6.

Chauhan, V. and Surgenor, B. (2015). A Comparative Study of Machine Vision Based Methods for Fault Detection in an Automated Assembly Machine. *Procedia Manufacturing*, 1, 416-428.

Cheng, S., Wei, Q., & Ye, Z. (2012). Fault Diagnosis System Based on Fuzzy-Inference. *Fuzzy Information and Engineering, 4*(1), 51-61.

Dalpiaz, G., & Rivola, A. (1997). Condition Monitoring and Diagnostics in Automatic Machines: Comparison of Vibration Analysis Techniques. *Mechanical Systems and Signal Processing, 11*(1), 53-73.

Davies, E. (2005). *Machine Vision: Theory, Algorithms, Practicalities* (3rd ed.). Amsterdam: Elsevier.

Davis, J. & Goadrich, M. (2006). The Relationship between Precision-Recall and ROC Curves. *Proc. of the 23rd international conference on Machine learning*, 233-240.

Demetgul, M., Tansel, I., & Taskin, S. (2009). Fault Diagnosis of Pneumatic Systems with Artificial Neural Network Algorithms. *Expert Systems with Applications, 36*(7), 10512-10519.

Demetgul, M., Unal, M., Tansel, I., & Yazicioğlu, O. (2011a). Fault Diagnosis on Bottle Filling Plant Using Genetic-Based Neural Network. *Advances in Engineering Software, 42*(12), 1051-1058.

Demetgül, M., Tansel, I., & Taskin, S. (2011b). Conditioning Monitoring and Fault Diagnosis for a Servo-pneumatic System with Artificial Neural Network Algorithms. *Artificial Neural Networks - Industrial and Control Engineering Applications, Prof. Kenji Suzuki (Ed.), InTech*, 441-458.

Egmont-Petersen, M., De Ridder, D., & Handels, H. (2002). Image Processing with Neural Networks—A Review. *Pattern Recognition, 35*(10), 2279-2301.

Fernando, H. (2014). Artificial Neural Networks for Fault Detection and Identification on an Automated Assembly Machine. *M. A. Sc. Thesis, Mechanical and Materials Engineering Department, Queen's University*, Kingston, ON.

FlyCaptureSDK. (Accessed: November 25, 2015). Retrieved from Point Grey Research Inc.: https://www.ptgrey.com/flycapture-sdk

Fujifilm USA. (Accessed: January 30, 2013). *Effective Focal Length, Lens Calculator*. Retrieved from http://www.fujifilmusa.com/products/optical_devices/machine-vision/lens-calculator/#effective-focal-length

Gonzalez, R. and Woods, R. (2002). *Digital Image Processing* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Groover, M. (2001). *Automation, Production Systems And Computer-Integrated Manufacturing* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Heath, M. D., Sarkar, S., Sanocki, T., & Bowyer, K. W. (1997). A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*(12), 1338-1359.

Holloway, L., & Krogh, B. (1990). Fault Detection and Diagnosis in Manufacturing Systems: A Behavioral Model Approach. *Proc. of Rensselaer's Second International Conference on Computer Integrated Manufacturing*, 252-259.

Horn, B. and Schunck, B. (1981). Determining Optical Flow. *Proc. International Society for Optics and Photonics, Technical Symposium East*, 319-331.

Huang, C., Lim, C., & Ming, C. (1992). Comparison of Image Processing Algorithms and Neural Networks in Machine Vision Inspection. *Computers & Industrial Engineering, 23*(1-4), 105-108.

Hughes, K., Szkilnyk, G., & Surgenor, B. (2012). A System for Providing Visual Feedback of Machine Faults. *Enabling Manufacturing Competitiveness and Economic Sustainability, springer*, 305-309.

Isermann, R. (2006). *Fault-Diagnosis Systems an Introduction From Fault Detection to Fault Tolerance.* Berlin: New York: Springer.

Kaewtrakulpong, P. and Bowden, R. (2001). An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection. *Proc. 2nd European Workshop on Advanced Video Based Survelliance Systems, AVBS01: Computer Vision and Distributed Processing*, 1-5.

Ke, Y., Sukthankar, R., & Hebert, M. (2007). Event Detection in Crowded Videos. *IEEE 11th International Conference on Computer Vision (ICCV)*, 1-8.

Killing, J., Surgenor, B., & Mechefske, C. (2009). A Machine Vision System for the Detection of Missing Fasteners on Steel Stampings. *The International Journal of Advanced Manufacturing Technology, 41*(7-8), 808-819.

Klein, R., Masad, E., Rudyk, E., & Winkler. (2014). Bearing Diagnostics using Image Processing Methods. *Mechanical Systems and Signal Processing, 45*(1), 105-113.

Konrad, H. (1996). Fault Detection in Milling, using Parameter Estimation and Classification Methods. *Control Engineering Practice, 4*(11), 1573-1578.

Kopparapu, S. K. (2006). Lighting Design for Machine Vision Application. *Image and Vision Computing, 24*(7), 720-726.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems,*1097-1105.

LabVIEW. (2010). *Vision Development Module*. National Instruments Corporation, U.S.A.

Lucas, B. D. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. *Proc. of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, 647-679.

Malamas, E., Petrakis, E., Zervakis, M., Petit, L., and Legat, J. (2003). A Survey on Industrial Vision Systems, Applications and Tools. *Image and Vision Computing, 21*(2), 171-188.

Malge, P. S., & Nadaf, R. S. (2014). PCB Defect Detection, Classification and Localization using Mathematical Morphology and Image Processing Tools. *International Journal of Computer Applications, 87*(9), 1015-1021.

Martínez, S. S., Ortega, J. G., García, J. G., & García, A. S. (2012). A Machine Vision System for Defect Characterization on Transparent Parts with Non-Plane Surfaces. *Machine Vision and Applications, 23*(1), 1-13.

MATLAB. (2014b). *Image Acquisition, Processing and Computer Vision Toolboxes.* The Mathworks, Inc. U.S.

McLachlan, G. and Peel, D. (2004). *Finite Mixture Models.* John Wiley & Sons Inc., New York.

Microscan Systems Inc. (Accessed: February 4, 2014). *NERLITE® Machine Vision Lighting*. Retrieved from http://www.microscan.com/enus/Technology/Machine%20Vision%20Lighting.aspx

Miles, B. C., Surgenor, B. W., and Killing, J. (2008). Effect of Lighting on the Performance of a Machine Vision System. *Proc. of 18th Conference on Flexible Automation and Intelligent Inspection (FAIM),* Skövde, Sweden.

National Instruments. (Accessed: January 25, 2014). Calculating Camera Sensor Resolution and Lens Focal Length, *Knowledge base: Image Acquisition*. Retrieved from http://digital.ni.com/public.nsf/allkb/29D716D6F4F1FBC386256AE700727AF6

Omid, M., Soltani, M., Dehrouyeh, M. H., Mohtasebi, S. S., & Ahmadi, H. (2013). An Expert Egg Grading System Based on Machine Vision and Artificial Intelligence Techniques. *Journal of Food Engineering, 118*(1), 70-77.

Patel, E., & Shukla, D. (2013). Comparison of Optical Flow Algorithms for Speed Determination of Moving Objects. *International Journal of Computer Applications, 63*(5), 32-37.

Patel, S., Kamrani, A., & Orady, E. (1995). A Knowledge-Based System for Fault Diagnosis and Maintenance of Advanced Automated Systems. *Computers & Industrial Engineering, 29*(1-4), 147-151.

Sekar, R., Hsieh, S., & Wu, Z. (2011). Remote Diagnosis Design for a PLC-Based Automated System: 1-Implementation of Three Levels of Architectures. *The International Journal of Advanced Manufacturing Technology, 57*(5-8), 683-700.

Sekar, R., Hsieh, S., & Wu, Z. (2013). Remote Diagnosis Design for a PLC-Based Automated System: 2-Evaluation of Factors Affecting Remote Diagnosis Performance. *The International Journal of Advanced Manufacturing Technology, 65*(5-8), 1091-1109.

Senthil Kumar, G., Natarajan, U., & Ananthan, S. (2012). Vision Inspection System for the Identification and Classification of Defects in MIG Welding Joints. *The International Journal of Advanced Manufacturing Technology, 61*(9), 923-933.

Shahabi, H., & Ratnam, M. (2009). In-Cycle Monitoring of Tool Nose Wear and Surface Roughness of Turned Parts using Machine Vision. *The International Journal of Advanced Manufacturing Technology, 40*(11), 1148-1157.

Shen, H., Li, S., Gu, D., & Chang, H. (2012). Bearing Defect Inspection Based on Machine Vision. *Measurement, 45*(4), 719-733.

Stauffer, C. and Grimson, W. E. (1999). Adaptive Background Mixture Models for Real-Time Tracking. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2, 1-7.

Sun, T. H., Tien, F. C., Tien, F. C., & Kuo, R. J. (2014). Automated Thermal Fuse Inspection using Machine Vision and Artificial Neural Networks. *Journal of Intelligent Manufacturing*, 1-13.

Sure controls Inc. (Accessed: March 25, 2016). *What is Industrial Automation ?* Retrieved from http://www.surecontrols.com/what-is-industrial-automation/

Szkilnyk, G. (2012). Vision-based Fault Detection in Assembly Automation. *M. A. Sc. Thesis, Mechanical and Materials Engineering Department, Queen's University*, Kingston, ON.

Szkilnyk, G., Hughes, K., Fernando, H., & Surgenor, B. (2012). Spatiotemporal Volume Video Event Detection for Fault Monitoring in Assembly Automation. *Proc. of 19th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, 20-25.

Usamentiaga, R., Molleda, J., Garcia, D., Bulnes, F. and Perez, J. (2013). Jam Detector for Steel Pickling Lines Using Machine Vision. *IEEE Transactions on Industry Applications, 49*(5), 1954-1961.

Viswanadham, N., & Johnson, T. (1988). Fault Detection and Diagnosis of Automated Manufacturing Systems. *Proc. of the 27th IEEE Conference on Decision and Control, 3*, 2301-2306.

Wang, W., & Mcfadden, P. (1996). Application of Wavelets to Gearbox Vibration Signals for Fault Detection. *Journal of Sound and Vibration, 192*(5), 927-939.

Wollak, E., & King, B. (2009). Success Through the Lens. *Intech Magazine (ISA), 56*(8), 16-19.

Xiaokun, L., & Porikli, F. (2004). A Hidden Markov Model Framework for Traffic Event Detection using Video Features. *Proc. of the International Conference on Image Processing (ICIP), 5*, 2901-2904.

Yao, A. (2005). Design and Implementation of Web-Based Diagnosis and Management System for an FMS. *The International Journal of Advanced Manufacturing Technology, 26*(11-12), 1379-1387.

Zezhi, C., Pears, N., Freeman, M., & Austin, J. (2014). A Gaussian Mixturemodel and Support Vector Machine Approach To Vehicle Type and Colour Classification. *Intelligent Transport Systems (IET), 8*(2), 135-144.

Zorcolo, A., Escobar-palafox, G., Gault, R., Scott, R., & Ridgway, K. (2011). Study of Lighting Solutions in Machine Vision Applications for Automated Assembly Operations. *IOP Conference Series: Materials Science and Engineering, 26*, 012-019.

Appendix A Hardware Specifications

A-1 Motor Controller Specifications (PowerFlex 4 AC Drive)

Providing users with powerful motor speed control, the Allen-Bradley® PowerFlex 4 and PowerFlex 40 AC drives are ideal for machine level speed control and provide the application versatility to meet the demands of global OEMs and end users requiring flexibility, space savings and ease of use.

Flexible Packaging and Mounting Options

- Installation can be a virtual snap using the DIN rail mounting feature on selected drives
- Flange mount drives are available to reduce overall enclosure size
- Zero-Stacking™ Drives allow for ambient temperatures up to 40 °C (104 °F), saving valuable panel space. 50 °C (122 °F) ambient temperatures are permitted with minimal spacing between drives

Easy to Start up and Operate

- Integral keypad features a 4 digit display and 10 additional LED indicators providing intuitive control
- The keypad, control keys and local potentiometer are active out of the box, simplifying start up
- The most commonly programmed parameters are grouped together for fast and easy start up

Versatile Programming and Network Solutions

- Integral RS485 communications let the drives be used in a multi-drop network configuration. A serial converter module provides connectivity to any controller that has the ability to initiate DF1 messaging
- DriveExplorer and DriveTools SP software can be used to program, monitor and control the drives
- A NEMA/UL Type 4X remote and NEMA 1 handheld LCD keypad provide additional programming and control flexibility, both featuring the popular CopyCat function

Premier Integration with PowerFlex Drives

For simplified AC drive start-up and reduced development time using the Allen-Bradley Logix control platform, we've integrated PowerFlex® AC drive configuration with RSLogix™ 5000 software. This single-software approach simplifies parameter and tag programming while still allowing stand-alone drive software tool use on the factory floor.

1. PowerFlex 4 AC Drive 0.2…3.7 kW; 0.25…5 Hp @ 120, 240, 480V
2. PowerFlex 40 AC Drive, 0.4…11 kW; 0.5…15 Hp @ 120, 240, 480, 600V (Product shown with DeviceNet option)
3. PowerFlex 4 and 40 Flange Mount Drives
4. PowerFlex 40 NEMA /UL Type 4X/12 (IP66/54) 0.4…3.7 kW 0.5…5 Hp (No additional accessory required for communication options)

PowerFlex 4 AC Drive

Designed with simplicity and space savings in mind, the PowerFlex 4 is:

- Ideal for applications with limited panel space
- An economical replacement for electromechanical devices or DC solutions

PowerFlex 40 AC Drive

The PowerFlex 40 AC drive shares all the same features and functionality of the PowerFlex 4 AC drive. In addition, the PowerFlex 40 AC drive features sensorless vector control and has additional I/O capability. Designed with application versatility and robust performance in mind, the PowerFlex 40 AC drives also feature:

- 0…10V or 4…20 mA (10-bit) analog output for feedback or as reference for other drives
- Timer, counter and StepLogic® functions can reduce hardware design cost, simplifying control schemes
- Two analog input channels, including PID capability, offer enhanced application flexibility
- Integral communications options including DeviceNet,™ ControlNet,™ EtherNet/IP™ and a variety of 3rd-party networks
- IP66, NEMA/UL Type 4X/12 (indoor) – for mounting directly in the product environment. Listed by UL to resist dust and dirt and survive high pressure water spray. Also certified by NSF to ensure conformity with international food equipment standards

PowerFlex 40 Configured Drives

PowerFlex 40 Configured Drives simplify installation and start up by allowing users to order drive packages that combine operator interface, control, communications and power options in pre-packaged assemblies. Offering a number of commonly requested pre-engineered options, as well as more complex custom-engineered combinations, this program provides a wide range of motor control options.

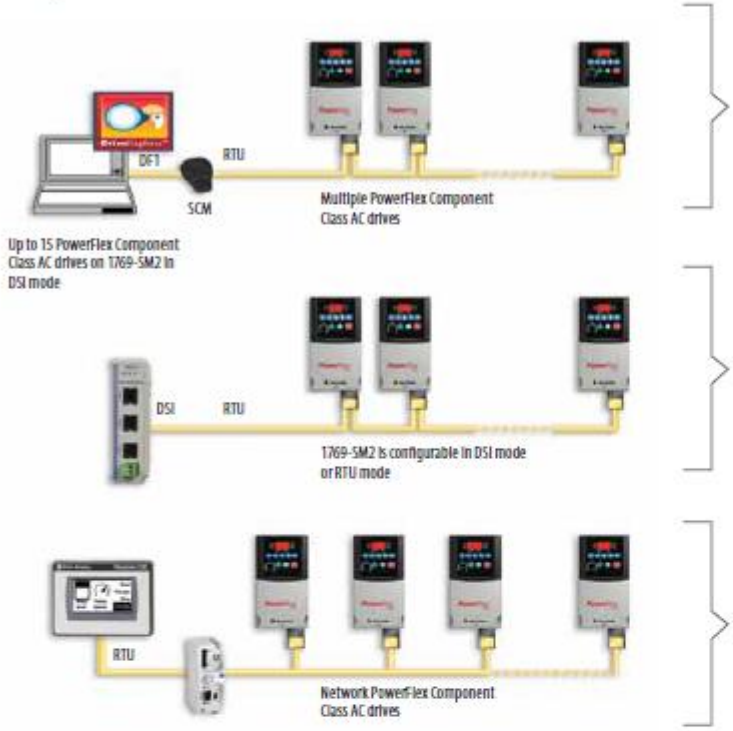
LISTEN. THINK. SOLVE.

Allen-Bradley · Rockwell Software

Rockwell Automation

Now we're talking…low cost communications

Simple RS485 Solutions

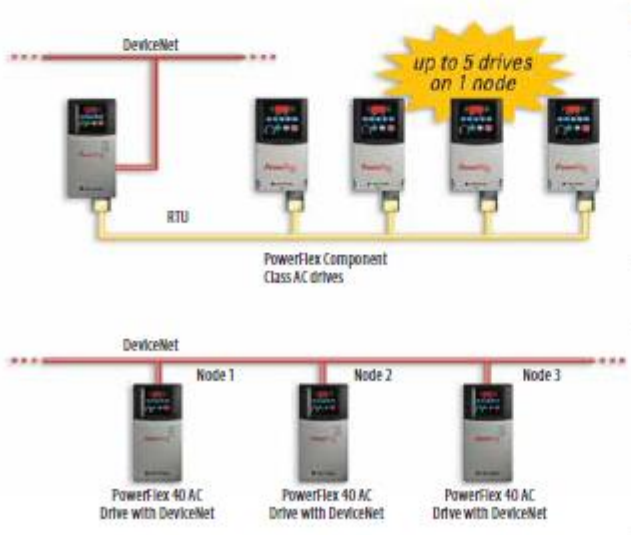


- Controls and monitors from a PC using DriveExplorer™ or DriveTools™ SP software
- Requires use of Serial Converter Module (SCM)

- Simple network control with any logic controller using DSI

- Compatible with any device that acts as RTU Master
- Drive supports standard 03 and 06 RTU commands

Advanced Network Solutions



- Multi-drive solution using a single PowerFlex 40 DeviceNet option
- Significantly reduces node count and system cost

- Network configuration using PowerFlex 40 drives with DeviceNet option cards
- Provides highest performance and most flexible configuration capabilities

and even lower cost machine level control

Timer and Counter Functions

- Digital inputs control digital outputs based on timer or counter function

Ideal for:

Mixers

Fillers

Shrink-wrap machines

Basic Logic Functions

- Digital inputs control digital outputs based on Boolean logic
- AND and OR logic inputs provide application flexibility

Ideal for:

Packaging machines

Conveyors

Palletizers

Step Logic Function

- Logic controlled steps using preset speed settings
- Each step can be programmed to:
  - Step based on digital input status including AND, OR and NOT logic
  - Step based on specific time
  - Control speed, direction and accel/decel rate
  - Control the status of an output
  - Make deterministic jumps

Ideal for:

Positioning

Shuttle transfer cars

Machine tool

Batch process

NEMA/UL Type 4X/12 (IP66/54)

- NEMA/UL Type 4X/12 for Washdown application Standards met:
  - UL Type 4X (high pressure water)
  - UL Type 12 (dust)
  - IP66/54 (dust and high pressure water)
  - C-Tick, CE
  - NSF
- Communication options do not require additional accessories

PowerFlex 40 NEMA/UL Type 4X/12 IP66/54

Ideal for washdown or dust-tight areas in:

Food and Beverage Applications

Pulp and Paper

Machine Tool

Textile

Pharmaceutical

A.2 PLC Specifications

MicroLogix 1400

Small Programmable Logic Controller

Allen-Bradley

Features and Benefits

- Expand your application capabilities with up to 7 expansion I/O modules for a maximum of 256 discrete I/O
- Up to 6 embedded 100 kHz highspeed counters (on controllers with dc inputs)
- 2 Serial ports with DF1/ DH485/ Modbus RTU/DNP3/ASCII protocol support
- Ethernet port provides you with EtherNet/IP, DNP3 over IP and Modbus TCP/IP protocol support as well as web server and email capabilities
- Built-in LCD with backlight allows you to view controller and I/O status, and provides a simple interface for messages, bit / integer monitoring and manipulation

Product Description

The Allen-Bradley® MicroLogix™ 1400 from Rockwell Automation complements the existing MicroLogix family of small programmable logic controllers. MicroLogix 1400 combines the features you demand from MicroLogix 1100, such as EtherNet/IP, online editing, and a built-in LCD, plus provides you with enhanced features, such as: higher I/O count, faster High Speed Counter/PTO and enhanced network capabilities

Take advantage of the built-in LCD with back lighting to set the Ethernet network configuration, display floating point values on a user configurable display, display OEM logos at startup and read or write any binary, integer and long file elements in the data table.

Three embedded communication ports provide you with excellent communications capabilities. MicroLogix 1400 offers an isolated RS232C/ RS485 combination port; a non-isolated RS232C port; and an RJ-45 port for 10/100 Mbps EtherNet/IP peer-to-peer messaging, DNP3 over IP and Modbus TCP/IP protocol.

Similar to the rest of the MicroLogix family, MicroLogix 1400 is programmed with RSLogix 500 programming software (Version 8.1 and above) as well as RSLogix Micro programming software.

LISTEN. THINK. SOLVE.

Allen-Bradley · Rockwell Software

Rockwell Automation

Product Specifications

MicroLogix	1766-L32BWA	1766-L32AWA	1766-L32BXB	1766-L32BWAA	1766-L32AWAA	1766-L32BXBA
Input Power	120/240 V AC		24V DC	120/240 V AC		24V DC
Memory	non-volatile battery backed RAM					
User Program / User Data Space	10K / 10K configurable					
Data Logging / Recipe Storage	128 K (without Recipe) / up to 64 K (after subtracting Data Logging)					
Battery Back-up	Yes					
Back-up Memory Module	Yes					
Digital Inputs	(12) Fast 24VDC (8) Normal 24VDC	(20) 120VAC	(12) Fast 24VDC (8) Normal 24VDC	(12) Fast 24VDC (8) Normal 24VDC	(20) 120VAC	(12) Fast 24VDC (8) Normal 24VDC
Digital Outputs	(12) Relay	(12) Relay	(6) Relay (3) Fast DC (3) Normal DC	(12) Relay	(12) Relay	(6) Relay (3) Fast DC (3) Normal DC
Analog Inputs / Outputs	None			(4) Voltage Inputs / (2) Voltage Outputs		
Serial Ports	(1) RS232C/RS485* , (1) RS232C**					
Serial Protocols	DF1 Full Duplex, DF1 Half Duplex Master/Slave, DF1 Radio Modem, DH-485, Modbus RTU Master/Slave, ASCII, DNP 3 Slave					
Ethernet Ports	(1) 10/100 EtherNet/IP port					
Ethernet Protocols	EtherNet/IP messaging, DNP3 over IP and Modbus TCP/IP					
Trim Potentiometers	2 Digital					
High-Speed Inputs	Up to 6 channels @ 100 kHz	N/A	Up to 6 channels @ 100 kHz	Up to 6 channels @ 100 kHz	N/A	Up to 6 channels @ 100 kHz
Real Time Clock	Yes, embedded					
PID	Yes (limited by loop and stack memory)					
PWM /PTO	N/A		3 channel PTO (100kHz)\PWM (40kHz)	N/A		3 channel PTO (100kHz)\PWM (40kHz)
Embedded LCD	Yes					
Floating Point Math	Yes					
Online Editing	Yes					
Operating Temperature	-20°C...+60°C					
Storage Temperature	-40°C (or -30°C)...+85°C					

* Isolated. RS232/RS485 combo port. Same as MicroLogix 1100 Comm 0
** Non-Isolated RS232. standard D-sub connector.

Rockwell Automation is an official ENERGY STAR® Industrial Service and Product Provider. It has proven it provides energy efficiency services and/or products to commercial buildings and industrial manufacturing plants in the United States by collaborating with an ENERGY STAR Industrial Partner to submit a learning profile that outlines the scope and resulting savings from energy efficiency-driven projects. For more information, visit ENERGY STAR for Industry at **www.energystar.gov/index.cfm?c=industry.bus_industry**

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters
Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

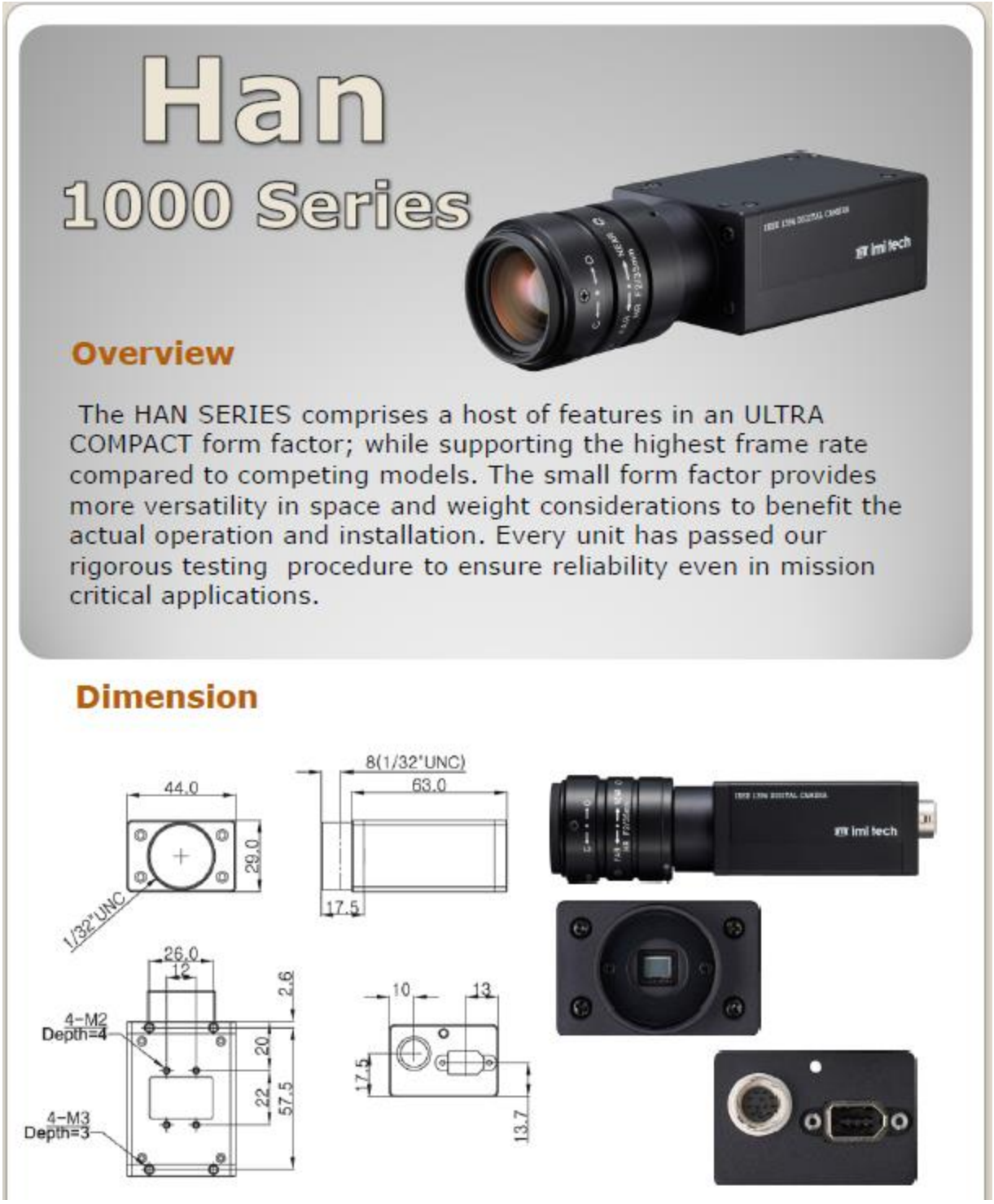
Publication 1766-PP001D-EN-P – April, 2011
Supercedes Publication 1766-PP001C-EN-P October, 2010

A.3 HMI C600 Specifications

2711C	C600 (T6T)	C600 (T6M)
	Touch Screen	Touch Screen
Display		
Display Description	Color transmissive TFT active matrix LCD	Monochrome transmissive FSTN passive matrix LCD
Display Size	5.7 inch	
Display Area (WxH)	115 x 86 mm (4.53 x 3.39 in.)	
Resolution	320 x 240	
Backlight	40,000 h min. White LED backlight life, not replaceable	50 000 h min. CCFL backlight life, not replaceable
Real-time Clock	Battery backup	
Touch Screen		
Touch Screen Description	Analog resistive	
Actuation Rating, Touch	1 000 000 presses	
Electrical		
Communication Ports	RS-232 (DH-485), RS-232 (DF1), RS485, Ethernet	
Programming Port	USB device port or Ethernet port	
Memory Card	USB flash drive and secure digital (SD) card	
Input Voltage Range	18...30V DC (24V DC nom.)	
Power Consumption, Max.	10 W max. (0.42 A at 24V DC)	
Environmental		
Operating Temperature	0...50 °C (32...122 °F)	
Nonoperating Temperature	-25...70 °C (-13...158 °F)	
Relative Humidity	0...95% noncondensing	
Operating Shock	15 g at 11 ms	
Nonoperating Shock	30 g at 11 ms	
Vibration	2 g at 10...500 Hz	
Enclosure Type Rating	Series B: NEMA/UL Type 4X (indoor), 12, 13, and IEC IP54, IP65 Series C: NEMA/UL Type 4X (indoor), 12, 13, and IEC IP54, IP65	
Certifications	c-UL-us, CE marked, C-Tick	
Weight, Aprox.		
Weight	0.68 kg (1.48 lb)	
Dimensions, Approx.		
Dimensions (HxWxD)	154 x 209 x 57 mm (6.0 x 8.23 x 2.25 in.)	
Dimensions, Cutout (HxW)	136 x 190 mm (5.35 x 7.48 in.)	

A.4 Camera Specifications

IEEE 1394 Digital Camera Specifications



Specifications

Model		IMx-1200FT	IMx-3145FT	IMx-1140FT	IMx-1080FT	IMx-1070FT	IMx-1050FT	IMx-1040FT
Image Sensor		1/1.8" Sony CCD	2/3" Sony CCD	1/2" Sony CCD	1/3" Sony CCD	1/2" Sony CCD	1/2" Sony CCD	1/3" Sony CCD
Sensor Name		ICX274 AL/AQ	ICX285 AL/AQ	ICX205 AL/AK	ICX204 AL/AK	ICX415 AL/AK	ICX414 AL/AK	ICX424 AL/AQ
Resolution	Megapixel	2.0MP	1.4MP	1.4MP	0.78MP	0.45MP	0.3MP	0.3MP
	HxV	1600 x 1200	1388 x 1040 (Mono) 1388 x 1036 (Color)	1392 x 1040 (Mono) 1392 x 1036(Color)	1032 x 776 (Mono) 1028 x 772 (Color)	782 x 582	656 x 484 (Mono) 652 x 480 (Color)	656 x 488 (Mono) 652 x 484 (Color)
Pixel Size		4.40μm x 4.40μm	6.45μm x 6.45μm	4.65μm x 4.65μm	4.65μm x 4.65μm	8.3μm x 8.3μm	9.90μm x 9.90μm	7.40μm x 7.40μm
Frame Rate		16 fps	20 fps	20 fps	30 fps	60 fps	86 fps	86 fps
Digital Interface		IEEE 1394a (12pin) 1 Port						
Transfer Rate		400 Mbps						
Scaning System		Progressive Scan						
Shutter Type		Global Shutter						
Data Path		12bit, Raw RGB, YUV422, Mono 8, Mono 16						
Binning		2x2, 1x2 (Mono only) * Color/Mono is available for Imx-1200FT						
ROI		Partial Scan (Unit:4x4)						
Trigger		External or Software Trigger with various mode						
I/O Port		Ext. Power, Ext. Trigger, RS232, Strobe (12-pin)						
Memory Save/Load		16 Channels (0:factory, 1~4:feature, 5~15:mode/feature)						
SIO(RS-232)		IIDC Ver.1.31 : Pass through or IMI-Tech Command						
Gain Control		0 ~ 18 dB (Manual or Auto)						
Shutter Speed		Manual or Auto / Range : 1 μsec ~ 3600 sec						
Control Functions		Brightness, Sharpness, Gamma, Auto-gain, Auto-Shutter Pan/Tilt, User defined AE, Hue, Saturation *Auto White Balance						
Lens Mount		C Mount						
Operating Temp.		-5°C ~ 45°C						
Storage Temp.		-30°C ~ 60°C						
Supply Voltage		DC 8V ~ DC 30V						
Dimension		44(W)x29(H)x63(D)mm * 44x29x67 for Imx-3145FT / 44x29x63 for Imx-1200FT						

Note : Specification may be subject to change without notice.

imi tech
(431-767) #616, Mega Valley, 799, Gwanyang 2-dong,
Dongan-gu, Anyang-si, Gyeonggi-do, Korea
TEL : +82-31-423-9801(rep), FAX : +82-31-423-9803,
http://www.imi-tech.com

USB 3.0 Camera Specifications

Grasshopper3 2.8 MP Color USB3 Vision (Sony ICX687) Specifications

Resolution	1928 x 1448
Frame Rate	26 FPS
Megapixels	2.8 MP
Chroma	Color
Sensor Name	Sony ICX687
Sensor Type	CCD
Readout Method	Global shutter
Sensor Format	1/1.8"
Pixel Size	3.69 µm
Lens Mount	C-mount
ADC	14-bit
Quantum Efficiency Blue (% at 470 nm)	50 (Mode 0) / 51 (Mode 7)
Quantum Efficiency Green (% at 525 nm)	55 (Mode 0) / 56 (Mode 7)
Quantum Efficiency Red (% at 640 nm)	49 (Mode 0) / 49 (Mode 7)
Temporal Dark Noise (e-)	10.17 (Mode 0) / 7.90 (Mode 7)
Absolute Sensitivity Threshold (y)	20.37 (Mode 0) 15.82 (Mode 7)
Saturation Capacity (e-)	8948 (Mode 0) / 8637 (Mode 7)
Dynamic Range (dB)	58.47 (Mode 0) / 60.24 (Mode 7)
Gain Range	-3.449 dB to 24 dB
Exposure Range	0.03 ms to 32 seconds
Trigger Modes	Standard, bulb, low smear, overlapped, multi-shot
Partial Image Modes	Pixel binning, ROI
Image Processing	Gamma, lookup table, hue, saturation, and sharpness
Image Buffer	128 MB
User Sets	2 user configuration sets for custom

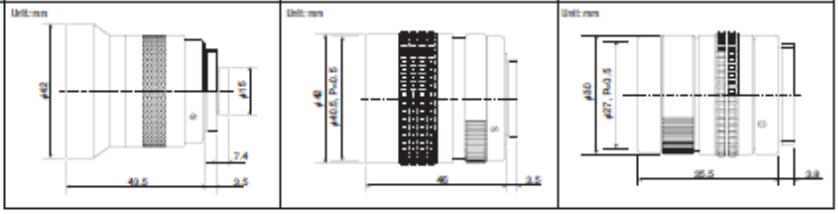
	camera settings
Flash Memory	2 MB non-volatile memory
Opto-isolated I/O Ports	1 input, 1 output
Non-isolated I/O Ports	2 bi-directional
Serial Port	1 (over non-isolated I/O)
Auxiliary Output	3.3 V, 150 mA maximum
Interface	USB 3.0
Power Requirements	5 V via USB 3.0 or 5-24 V via GPIO (external power is recommended for this model)
Power Consumption (Maximum)	4.5 W
Dimensions	44 mm x 29 mm x 58 mm
Mass	90 grams
Machine Vision Standard	USB3 Vision v1.0
Compliance	CE, FCC, KCC, RoHS
Temperature (Operating)	0° to 50°C
Temperature (Storage)	-30° to 60°C
Humidity (Operating)	20 to 80% (no condensation)
Humidity (Storage)	20 to 95% (no condensation)
Warranty	3 years

A.5 Lens Specifications

PENTAX

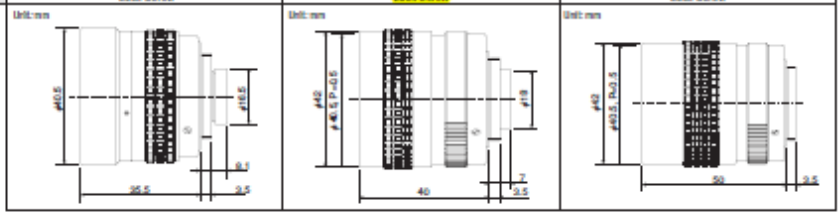
Monofocal Manual Iris Lens

Model		H416 (C60402)	H612A (C60607)	H1212B (C61215)
Format Size		1/2	1/2	1/2
Focal Length		4.2mm	6.0mm	12.0mm
Max. Aperture Ratio		1:1.6	1:1.2	1:1.2
Iris Range		1.6–C	1.2–C	1.2–22
Mount		C	C	C
Horizontal Angle of View	1/4 format	47.87°	32.97°	16.93°
	1/3 format	64.27°	43.55°	22.60°
	1/2 format	86.77°	56.93°	30.18°
	2/3 format	—	—	—
Min. Object Distance		0.2m	0.2m	0.2m
Back Focal Length		10.42mm	14.31mm	13.87mm
Filter Size		—	40.5mm P=0.5mm	27.0mm P=0.5mm
Dimensions		42×45.5mm	42×46mm	30×35.5mm
Weight		120g	125g	67g
Remarks		Lock Screw	Lock Screw	Lock Screw



Monofocal Manual Iris Lens

Model		C418DX (C30405)	C815B (C30811)	B1214D-2 (C21211)
Format Size		2/3	2/3	1
Focal Length		4.8mm	8.5mm	12.5mm
Max. Aperture Ratio		1:1.8	1:1.5	1:1.4
Iris Range		1.8–C	1.5–C	1.4–C
Mount		C	C	C
Horizontal Angle of View	1/4 format	41.68°	24.02°	16.21°
	1/3 format	55.11°	31.87°	21.53°
	1/2 format	72.37°	42.00°	28.51°
	2/3 format	96.36°	56.40°	38.65°
Min. Object Distance		0.3m	0.2m	0.3m
Back Focal Length		9.71mm	10.88mm	14.40mm
Filter Size		—	40.5mm P=0.5mm	40.5mm P=0.5mm
Dimensions		40.5×35.5mm	42×40mm	42×50mm
Weight		105g	120g	135g
Remarks		Lock Screw	Lock Screw	Lock Screw



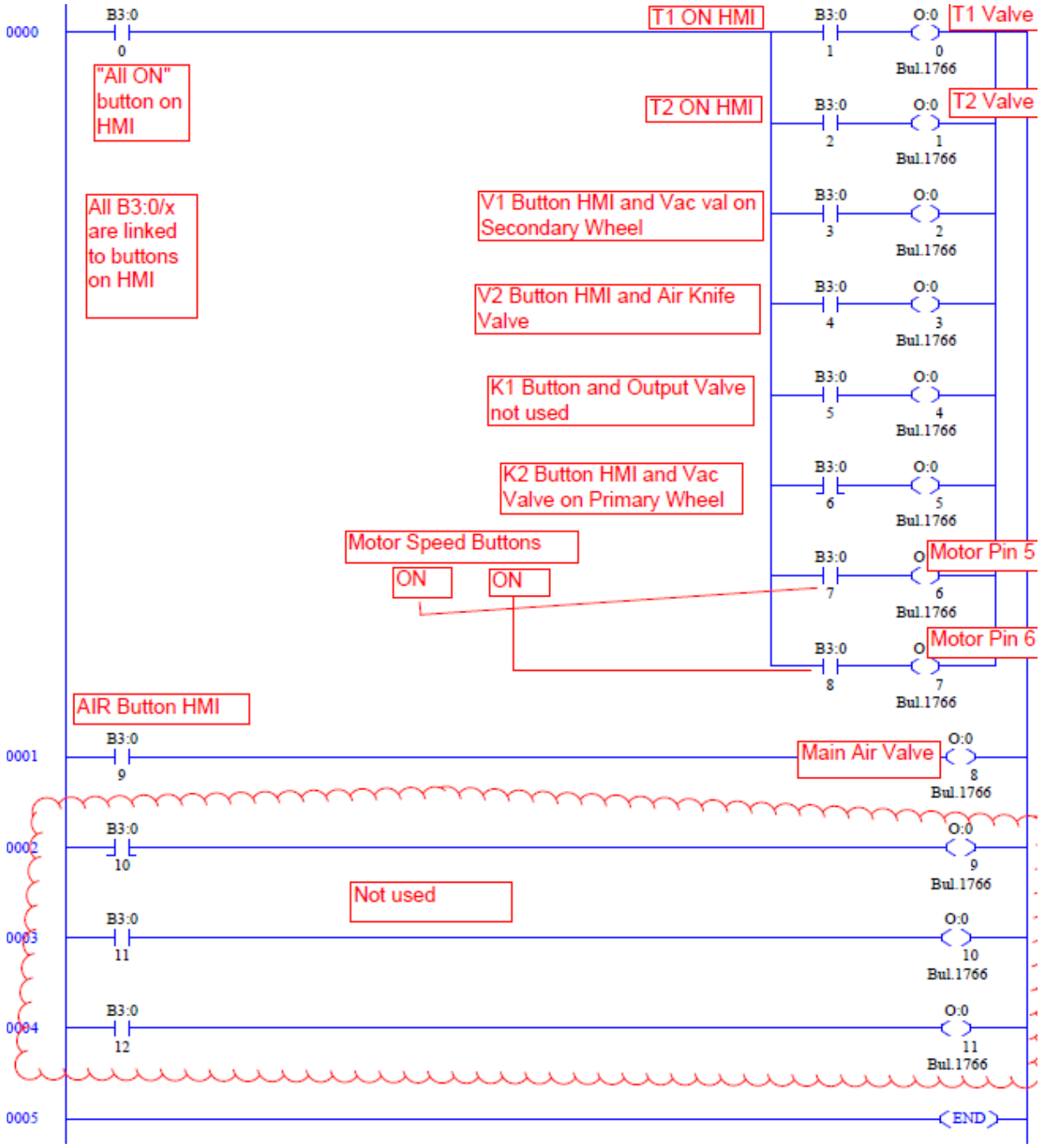
A.6 Light Specifications

Aputure Amaran AL-528W Daylight LED Flood Light Specifications

Light Source	528 LEDs, 100,000 hr life expectancy
Beam Angle	75°
Color Temperature	5,500 K
Brightness	4,040 lux/374 fc @ 1.6' (0.5 m) 1,220 lux/113 fc @ 3.3' (1.0 m)
CRI	88
Battery Power	2 x Sony F/FM/QM series or similar (optional)
Power	Multi-voltage AC 18 VDC
Operating Current	2.8 A
Power Consumption	30 W
Cooling	Convection
Average Life Hours	100,000
Dimensions	9.37 x 7.48 x 1.26" (23.8 x 19.0 x 3.2 cm)
Weight	1.19 lb (540 g)

Appendix B PLC Program

Ladder Diagram for Machine and Pneumatic System Control



Appendix C MATLAB Programs

There were more than 40 pages of MATLAB programs written for the three methods. Moreover, programs also required video files as inputs. The size of video data was more than 10 GB. Therefore, MATLAB programs written for video data acquisition and fault detection and classification algorithms have been archived on a USB Flash Drive. The explanation of which blocks of the programs are author's own and which block of the programs are predefined function in MATLAB is given in the flow charts of the corresponding methods (Figure 4-16 for Method 1, Figure 4-26 for Method 2 and Figure 4-33 for Method 3) in Chapter 4.

C.1 Programs on a USB Flash Drive

MATLAB programs written for video data acquisition and fault detection and classification algorithms have been archived on a USB Flash Drive. The list of folders and files on the USB is given below. The USB has a main folder with the name: USB for Vedang's Thesis Programs. All other folders are subfolders of the main folder. The folder tree structure is shown in the Figure C-1. "O-ring Machine Movie" folder contains a video of the machine's operation sequence.

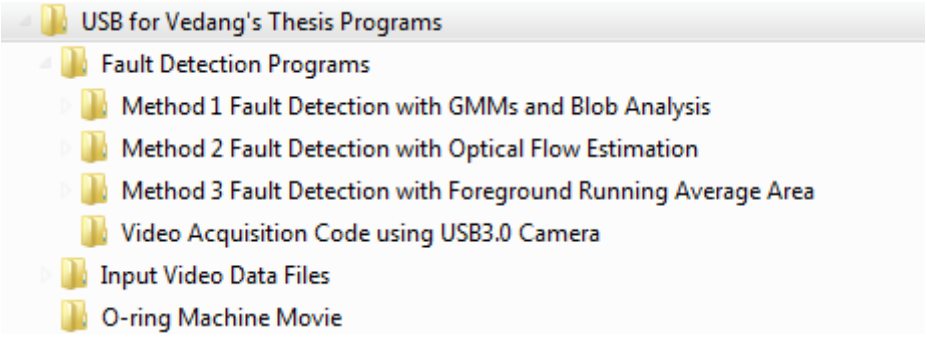


Figure C-1 Folder structure on the USB

C.2 Program Requirements

The Programs are written in MATLAB. Hence, to use them the user should have MATLAB version 2013 or higher with the following toolboxes installed.

- Image Acquisition Toolbox
- Image Processing Toolbox
- Computer Vision Toolbox

There are three MVI based fault detection and classification methods developed in the thesis. Therefore, programs are available for all three methods. They all have similar GUIs but the content within the program are written for the specific method. The steps to use the programs are the same. Hence, the procedure is

explained only for the Method 1 Programs. The other methods have the same procedure with the difference of only change in the respective folder as a current working directory of MATLAB.

C.3 How to Use Programs

The steps to use Method 1: Fault Detection with GMMs and Blob Analysis programs are explained below.

1. Connect the USB and start MATLAB
2. Set Path for the USB's main folder as shown in Figure C-2 and Figure C-3. Save and Close.

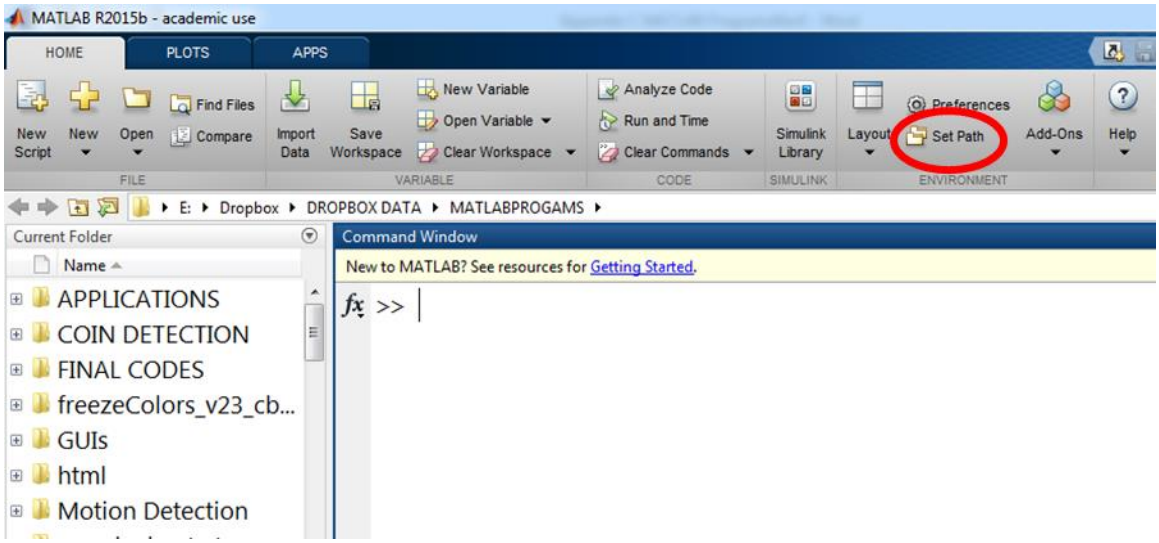


Figure C-2 Set path for the USB's Folders

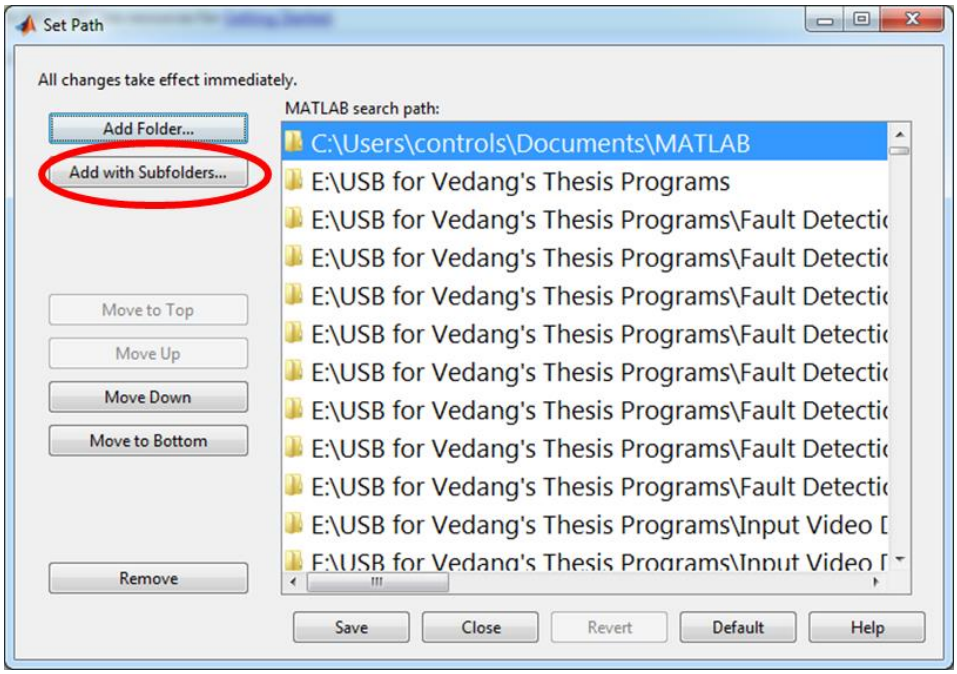


Figure C-3 Add with subfolders

3. Change MATLAB's current directory to : … \USB for Vedang's Thesis Programs\Fault Detection Programs\Method 1 Fault Detection with GMMs and Blob Analysis, as shown in Figure C-4.

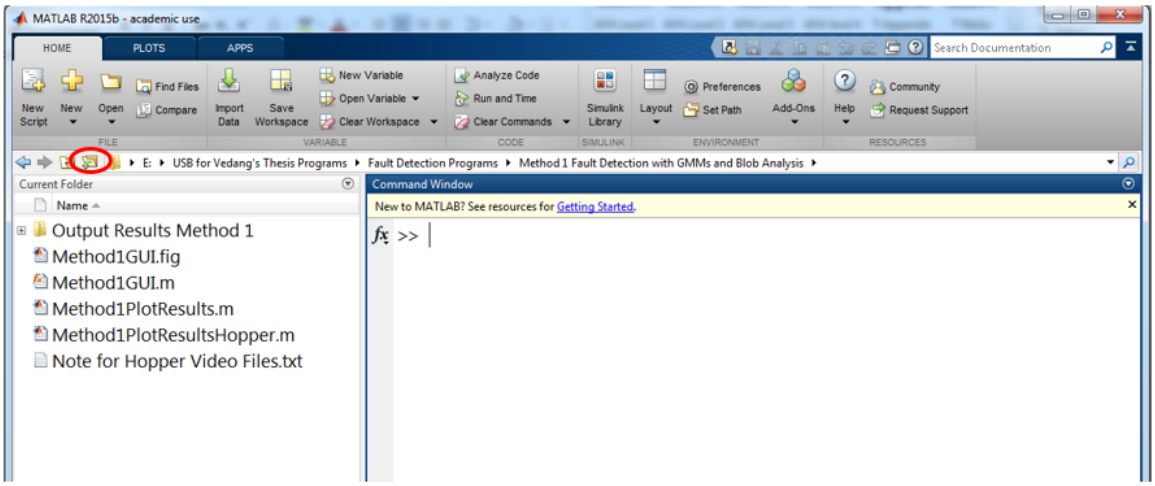


Figure C-4 Change current working directory

4. Type command guide in the command window. It opens a GUIDE Quick Start window. Select the tab Open Existing GUI, click Browse and open Method1GUI.

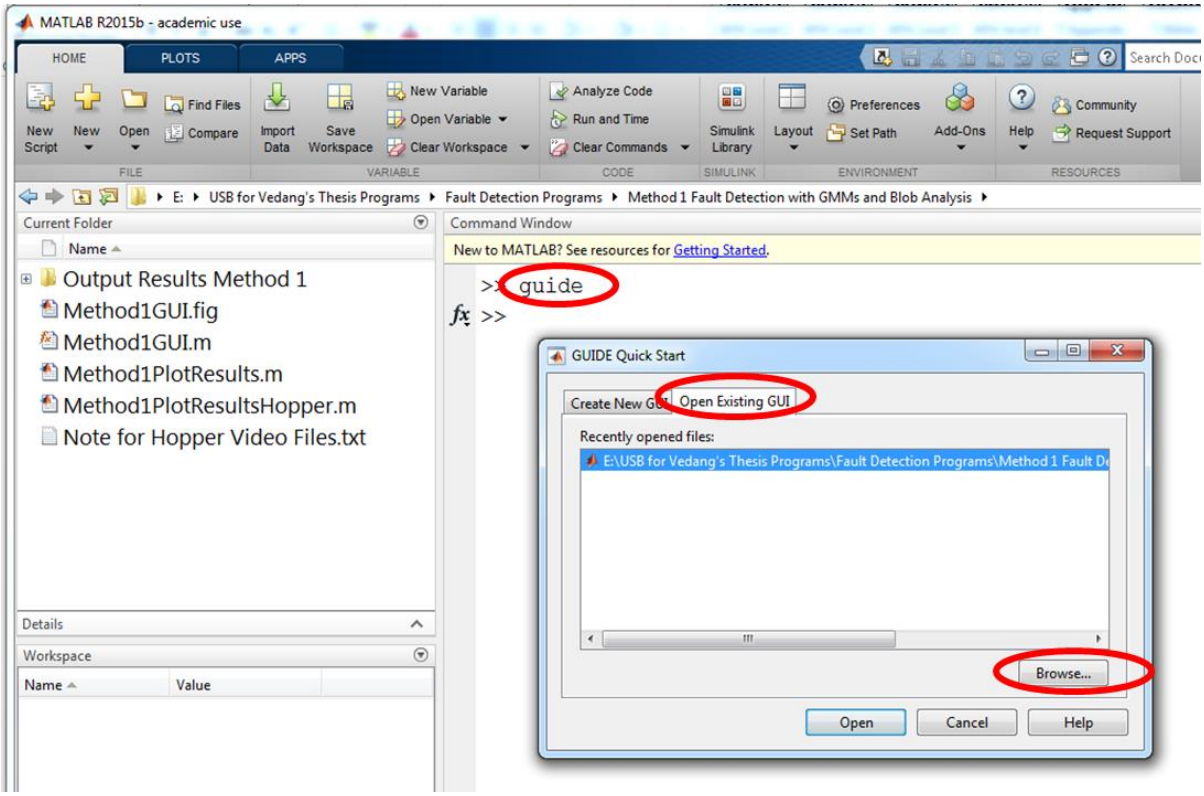


Figure C-5 Open Existing GUI

5. Once Method1GUI opens. Click Run (green) button as shown in Figure C-6.

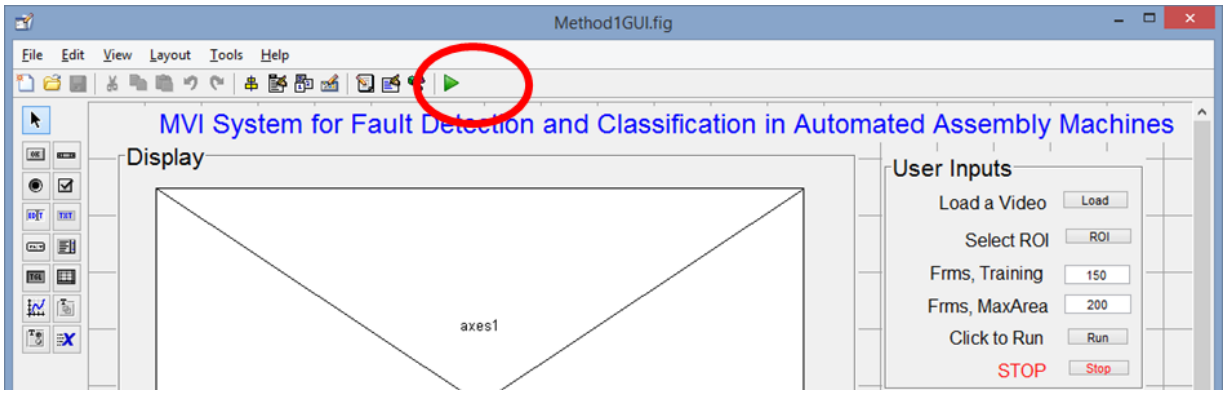


Figure C-6 Run Method 1 GUI

6. The GUI code is working now. Provide the video file path (from the USB folder "Input Video Data Files") for one of the ROIs video. Click load button. Set the appropriate ROI from the drop-down list and click the Run button as shown in Figure C-7.

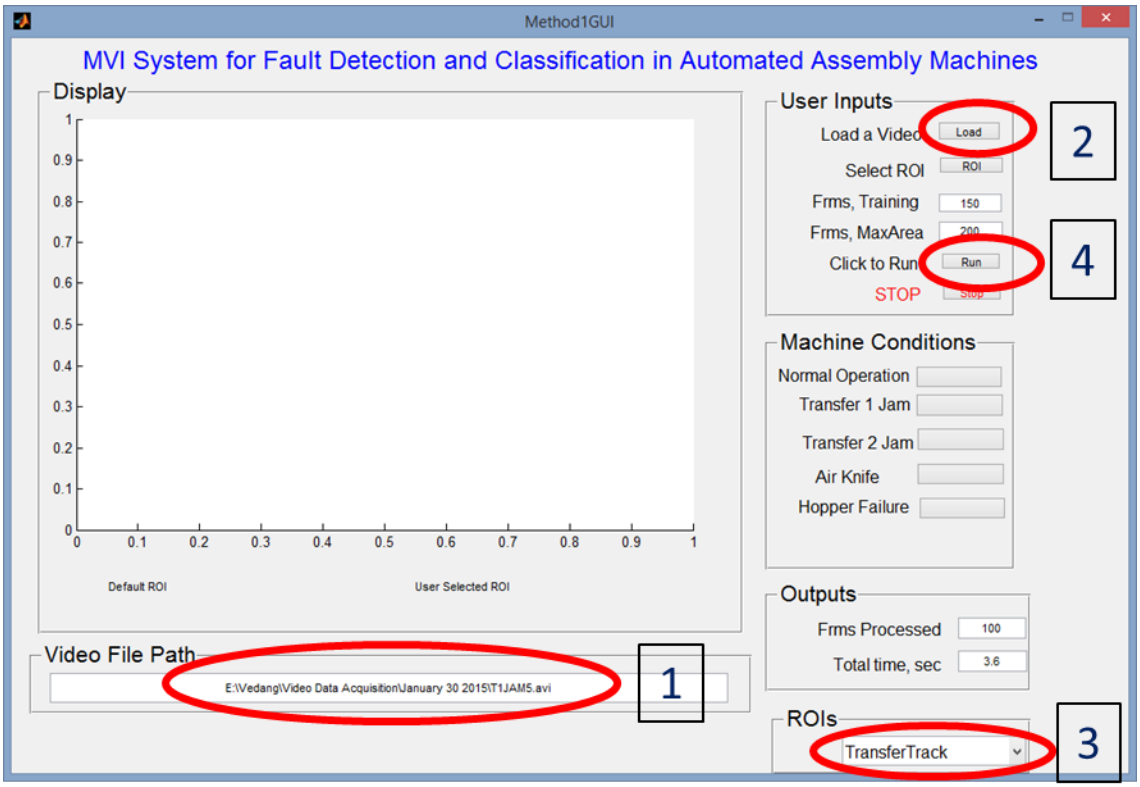


Figure C-7 Perform steps 1 to 4

7. Get the plot and a frame as shown in Figure C-8.

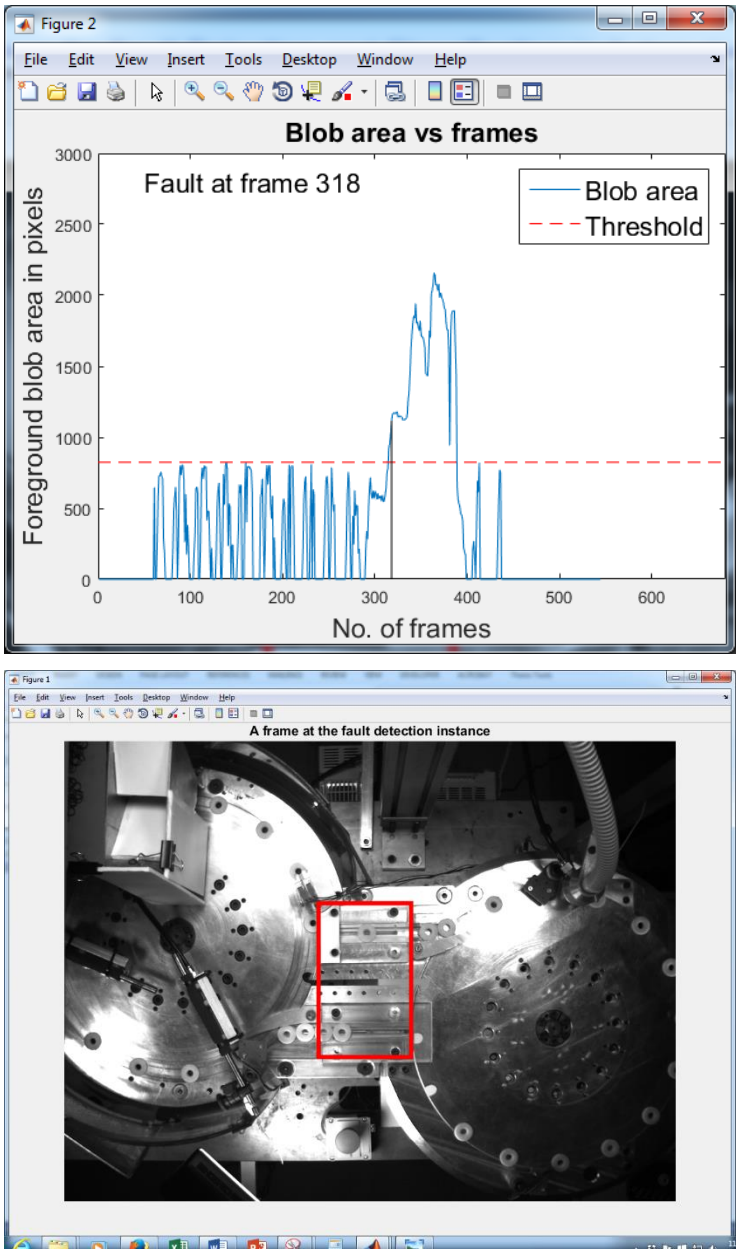


Figure C-8 Program output for T1JAM5.avi

8. The output results (the frame and plot) are also stored on the USB in "Output Results Method 1" folder. This folder is a subfolder of "Method 1 Fault Detection with GMMs and Blob Analysis".
9. For the hopper video files to overwrite the default plot settings, run "Method1PlotResultsHopper.m" from the current working directory.
10. Follow the steps 1 to 9 to use Method 2 and Method 3 programs. Change MATLAB's current directory to the respective folder first.

Appendix D Performance Measurement Data Table

Table D-1 Fault detection frame numbers and processing times

Sr. No.	Video file	Frame no. @ fault (Total frames)	Fault detection frame no. @ fault detection (delay w.r.t. fault introduction frame no.)			Time to process 10 s long video file (300 frames)		
			Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
Transfer track ROI								
1	NO1	582	Pass	Fail	Pass	3.43	4.19	6.11
2	NO2	627	Pass	Fail	Pass	3.45	4.15	6.17
3	NO3	627	Pass	Fail	Pass	3.43	4.16	6.10
4	NO4	627	Pass	Fail	Pass	3.21	4.16	5.69
5	NO5	584	Pass	Pass	Pass	3.41	4.21	6.06
6	T1JAM1	300 (546)	348 (48)	420 (120)	357 (57)	3.42	4.19	6.10
7	T1JAM2	300 (545)	324 (24)	324 (24)	332 (32)	3.50	4.18	6.11
8	T1JAM3	248 (499)	269 (21)	290 (42)	277 (29)	3.50	4.18	6.12
9	T1JAM4	289 (544)	316 (27)	310 (21)	323 (34)	3.43	4.18	6.12
10	T1JAM5	297 (544)	318 (21)	389 (92)	326 (29)	3.40	4.18	6.10
11	T2JAM1	252 (500)	280 (28)	Fail	275 (23)	3.46	4.20	6.10
12	T2JAM2	390 (628)	418 (28)	Fail	424 (34)	3.42	4.20	6.09
13	T2JAM3	320 (628)	342 (22)	333 (13)	342 (22)	3.43	4.19	6.11
14	T2JAM4	342 (629)	357 (15)	Fail	367 (25)	3.41	4.21	6.11
15	T2JAM5	357 (628)	400 (43)	Fail	395 (38)	3.39	4.20	6.10
Air Knife ROI								
16	AirNO1	501	Pass	Pass	Pass	3.08	3.14	4.30
17	AirNO2	503	Pass	Pass	Pass	3.12	3.17	4.31
18	AirNO3	503	Pass	Pass	Pass	3.10	3.16	4.30
19	AirNO4	501	Pass	Pass	Pass	3.11	3.17	4.28
20	AirNO5	503	Pass	Pass	Pass	3.10	3.17	4.31
21	Air knife fault 1	290 (539)	313 (23)	310 (20)	324 (34)	3.12	3.17	4.29
22	Air knife fault 2	275 (627)	305 (30)	290 (15)	302 (27)	3.09	3.13	4.28
23	Air knife fault 3	353 (671)	428 (75)	370 (17)	429 (76)	3.09	3.13	4.29
24	Air knife fault 4	205 (587)	233 (28)	232 (27)	281 (76)	3.09	3.15	4.29
25	Air knife fault 5	238 (623)	307 (69)	280 (42)	260 (22)	3.09	3.13	4.27
Hopper ROI								
26	Hopper NO1	623	Pass	Fail	Pass	3.46	4.12	6.42
27	Hopper NO2	609	Pass	Fail	Pass	3.41	3.96	6.41
28	Hopper NO3	610	Pass	Fail	Pass	3.43	3.96	6.40
29	Hopper NO4	549	Pass	Fail	Fail	3.46	3.95	6.38
30	Hopper NO5	561	Pass	Pass	Pass	3.45	3.95	6.37
31	Hopper fault 1	381 (547)	434 (53)	Fail	Fail	3.49	3.93	6.27

32	Hopper fault 2	230 (560)	266 (36)	278 (48)	302 (72)	3.36	3.91	6.27
33	Hopper fault 3	330 (609)	368 (38)	382 (52)	399 (69)	3.40	3.93	6.28
34	Hopper fault 4	320 (623)	363 (43)	356 (36)	403 (83)	3.42	3.93	6.28
35	Hopper fault 5	307 (609)	350 (43)	Fail	335 (28)	3.41	3.97	6.30
	Average	**577**	**36**	**35**	**43**	**3.33**	**3.83**	**5.64**
	Std. deviation	50.83	16.06	21.64	21.55	0.15	0.44	0.86
	Std. error	8.59	2.71	3.66	3.64	0.03	0.07	0.15

Need for Dynamic Threshold

Table D-2 lists dynamically estimated threshold values from first 200 frames (1 rotation of the wheels) of each video file. The variations were observed between the thresholds three ROIs. Moreover, there was a significant variation in the threshold values for the five video files of the same ROI. For example, Method 1 had highest 24 % standard deviation in the threshold for the air knife ROI. Method 2 had highest 58 % standard deviation in the threshold for the hopper ROI and Method 3 had highest 26 % standard deviation in for the hopper ROI. This variations justified the need for dynamic estimation of the threshold values.

Table D-2 Dynamically calculated threshold values

Sr. No.	**Video file**	**Dynamic threshold estimated from first 200 frames (1 rotation of the wheel) of video files**		
		Method 1	**Method 2**	**Method 3**
		Transfer track ROI		
1	NO1	836	31.06	901.3
2	NO2	844	31.41	891.1
3	NO3	833	30.96	888
4	NO4	826	32.06	900.6
5	NO5	838	37.25	889.7
6	T1JAM1	765	51.18	743.7
7	T1JAM2	785	50.47	765.4
8	T1JAM3	774	50.42	734
9	T1JAM4	834	49.13	875.4
10	T1JAM5	824	53.45	766
11	T2JAM1	816	51.54	756.7
12	T2JAM2	850	49.42	963.4
13	T2JAM3	816	53.26	721.6
14	T2JAM4	800	52.68	735.7
15	T2JAM5	835	55.32	731.3
	Average	818.40	45.31	817.59
	Std. deviation	25.99	9.58	84.08
	% Std. deviation	**3.18**	**21.14**	**10.28**

		Air knife ROI		
16	AirNO1	344	12.2	1504
17	AirNO2	400	12.03	1598
18	AirNO3	326	12.66	1521
19	AirNO4	206	12.73	1466
20	AirNO5	396	12.07	1548
21	Air knife fault 1	300	11.23	1564
22	Air knife fault 2	231	12.34	1464
23	Air knife fault 3	325	12.52	1618
24	Air knife fault 4	309	13.41	1591
25	Air knife fault 5	200	13.54	1445
	Average	303.70	12.47	1531.90
	Std. deviation	71.49	0.68	61.43
	% Std. deviation	**23.54**	**5.42**	**4.01**
		Hopper ROI		
26	Hopper NO1	160	2.68	462.7
27	Hopper NO2	148	1.69	478.6
28	Hopper NO3	173	2.31	447.9
29	Hopper NO4	175	1.98	326.8
30	Hopper NO5	193	5.6	417.9
31	Hopper fault 1	163	4.09	225
32	Hopper fault 2	202	8.24	652.9
33	Hopper fault 3	189	7.73	429.7
34	Hopper fault 4	179	6.49	554.9
35	Hopper fault 5	198	2.43	447
	Average	178.00	4.32	444.34
	Std. deviation	17.66	2.50	115.51
	% Std. deviation	**9.92**	**57.73**	**26.00**

Appendix E Tuned Parameters

MVI methods are application specific. This means, a MVI system is developed for one application and is when applied to a different application, the parameters of the method need to be retuned.

The procedure followed to obtain final tuned parameters is explained below.

- During the first stage of tuning, only transfer track ROI was considered and the parameter were tuned to successfully classify the operations of the machine into their correct categories. Transfer track ROI had moving objects (carriers and O-rings) on stationary background. Therefore, less number of frames were required for background estimation.
- Only air knife ROI was processed in the second stage of the algorithm development. Air knife ROI had moving objects (carriers and O-rings) on moving background (rotating wheel). Therefore, more number of frames were required for background estimation.
- Only hopper ROI was added in the third stage and the parameters were tuned to classify hopper operations into their correct categories.
- In the fourth stage, the parameters were tuned to successfully classify all (or maximum) ROIs operations using a fixed set of parameters for all video files. Table E-1 lists final tuned parameters for each MVI-based method. This was the basis for the results reported in Chapter 5.

Table E-1 Tuned parameters for all three MVI methods

	Method 1	Method 2	Method 3
1	No. of frames used for background estimation **150 frames**	Optical flow components type **Complex numbers**	Running average filter size for background estimation **20 frames**
2	Initial variance for GMMs **(30/255) * (30/255) pixels**	Subsampling size for optical flow field **Every 3rd component from optical flow lines**	Filter size for running average area calculation **7 elements in a vector**
3	Minimum blob area to be detected **160 pixels**	Scaling of flow fields **10 times**	No. of frames used to estimate a threshold **200 frames**
4	Maximum blob area **1600 pixels**	No. of frames used to estimate a threshold **200 frames**	
5	No. of frames used to estimate a threshold **200 frames**		

Appendix F Uncertainty Analysis in MVPI

Sn. No.	Machine Condition	Frame	Frame, fault detected			Delay, frames, in detection			Processing time, s ,with display		
		fault introduced	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3	Method 1	Method 2	Method 3
1	Normal 1	582	p	f	p				3.4348	4.1875	6.1114
2	Normal 2	627	p	f	p				3.4458	4.1550	6.1709
3	Normal 3	627	p	f	p				3.4312	4.1569	6.1001
4	Normal 4	627	p	f	p				3.2120	4.1566	5.6927
5	Normal 5	584	p	p	p				3.4096	4.2068	6.0616
6	T 1 Jam 1	300	348	420	357	48		57	3.4166	4.1872	6.0985
7	T 1 Jam 2	300	324	324	332	24	24	32	3.5006	4.1795	6.1117
8	T 1 Jam 3	248	269	290	277	21	42	29	3.4987	4.1840	6.1186
9	T 1 Jam 4	289	316	310	323	27	21	34	3.4320	4.1819	6.1198
10	T 1 Jam 5	297	318	389	326	21	92	29	3.4046	4.1807	6.0981
11	T2 Jam 1	252	280	500	275	28		23	3.4595	4.1971	6.1003
12	T2 Jam 2	390	418	628	424	28		34	3.4158	4.1983	6.0924
13	T2 Jam 3	320	342	333	342	22	13	22	3.4328	4.1864	6.1061
14	T2 Jam 4	342	357	629	367	15		25	3.4109	4.2070	6.1142
15	T2 Jam 5	357	400	628	395	43		38	3.3915	4.1983	6.0956
16	Air Knife Normal 1	501	p	p	p				3.0801	3.1441	4.2974
17	Air Knife Normal 2	503	p	p	p				3.1158	3.1655	4.3090
18	Air Knife Normal 3	503	p	p	p				3.0960	3.1578	4.3030
19	Air Knife Normal 4	501	p	p	p				3.1143	3.1672	4.2830
20	Air Knife Normal 5	503	p	p	p				3.0989	3.1721	4.3075
21	Air Knife Failure 1	290	313	310	324	23	20	34	3.1201	3.1739	4.2894
22	Air Knife Failure 2	275	305	290	302	30	15	27	3.0913	3.1317	4.2751
23	Air Knife Failure 3	353	428	370	429	75	17	76	3.0862	3.1285	4.2908
24	Air Knife Failure 4	205	233	232	281	28	27	76	3.0903	3.1515	4.2866
25	Air Knife Failure 5	238	307	280	260	69	42	22	3.0929	3.1337	4.2670
26	Hopper Normal 1	623	p	f	p				3.4550	4.1180	6.4151
27	Hopper Normal 2	609	p	f	p				3.4137	3.9609	6.4086
28	Hopper Normal 3	610	p	f	p				3.4278	3.9612	6.4022
29	Hopper Normal 4	549	p	f	f				3.4608	3.9530	6.3775
30	Hopper Normal 5	561	p	p	p				3.4489	3.9467	6.3701
31	Hopper Failure 1	381	434	547	547	53			3.4867	3.9305	6.2678
32	Hopper Failure 2	230	266	278	302	36	48	72	3.3646	3.9123	6.2708
33	Hopper Failure 3	330	368	382	399	38	52	69	3.4038	3.9347	6.2765
34	Hopper Failure 4	320	363	356	403	43	36	83	3.4184	3.9264	6.2784
35	Hopper Failure 5	307	350	609	335	43		28	3.4092	3.9696	6.2975
	Average	415.26	336.95	405.25	350.00	**35.75**	**34.54**	**42.63**	3.33	3.83	5.64
	Std. dev.	142.91	55.14	134.36	68.49	**16.06**	**21.64**	**21.55**	0.16	0.44	0.88
	Std. error	24.16	9.32	22.71	11.58	**2.71**	**3.66**	**3.64**	0.03	0.07	0.15

Uncertainty at 95 % confidence level = t*Std.error, (where t=2 is the correction factor from t distribution for finite samples)

Calculation for Uncertainty			
Related to Parameter P2	Method 1	Method 2	Method 3
Avg. Time to process 1 frame	3.33	3.83	5.64
Standard error	0.03	0.07	0.15
Uncertainty at 95 %	0.06	0.14	0.30
parameter p2	0.90	0.78	0.53
Uncertainty in p2 at 95%	**0.02**	**0.03**	**0.03**
Related to Parameter P3	Method 1	Method 2	Method 3
Avg. frames delay	36	35	43
Standard error	2.71	3.66	3.64
Uncertainty at 95 %	5.42	7.32	7.28
Parameter p3	0.80	0.83	0.57
Uncertainty in p3 at 95%	**0.18**	**0.24**	**0.24**

MVPI Calculation

	P1	P2	P3	P4	P5	W1	W2	**Overall**	Comp. 2	Comp. 3
Method 1	1	0.9	0.8	0.54	0.4	100	50	**77**	31	21
Method 2	0.6	0.78	0.83	0.24	0.5	100	50	**59**		
Method 3	0.94	0.53	0.57	0.14	0.7	100	50	**64**	8	

Propagation of Uncertainty in MVPI:

Uncertainty in MVPI is the root sum of the squares of individual uncertainty because the MVPI is the fuction of sum of p1 to p5

Only p2 and p3 have uncertainty	**Method 1**	**Method 2**	**Method 3**
Wb/3 (constant)	16.67	16.67	16.67
Uncertainty =	3.02	4.09	4.07
Overall MVPI at 95 % confidence	**77 ± 3.02**	**59 ± 4.09**	**64 ± 4.07**

Appendix G Lights Calibration Procedure

MVI methods are sensitive to lighting changes. The four LED panel lights were used for video data acquisition as shown in Figure G-1. The procedure of adding the four lights is given step by step in this appendix. The effect of the different lights on the appearance of images is also shown with the examples. The detection of O-rings required a light beneath the wheels. This backlighting effect was obtained using the white foam board underneath the secondary wheel. The effect is demonstrated by the images with and without the white foam board underneath the secondary wheel.

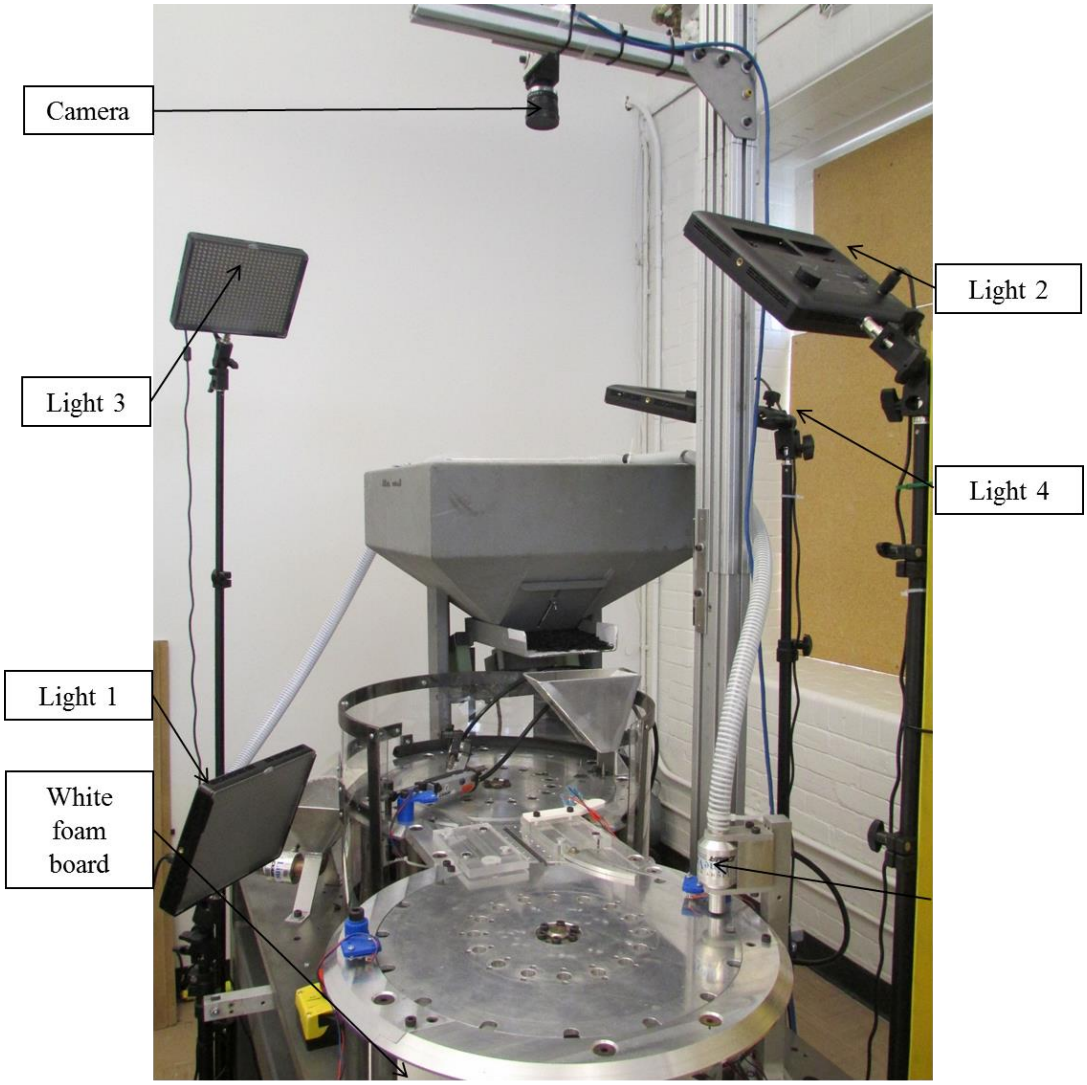


Figure G-1 Position of the four lights (Figure 3-2 is repeated for consistency)

1) Effect of only Light 1 on image appearance:

Figure G-2 shows that Light 1 illuminates only transfer track 1 ROI (lower middle region).

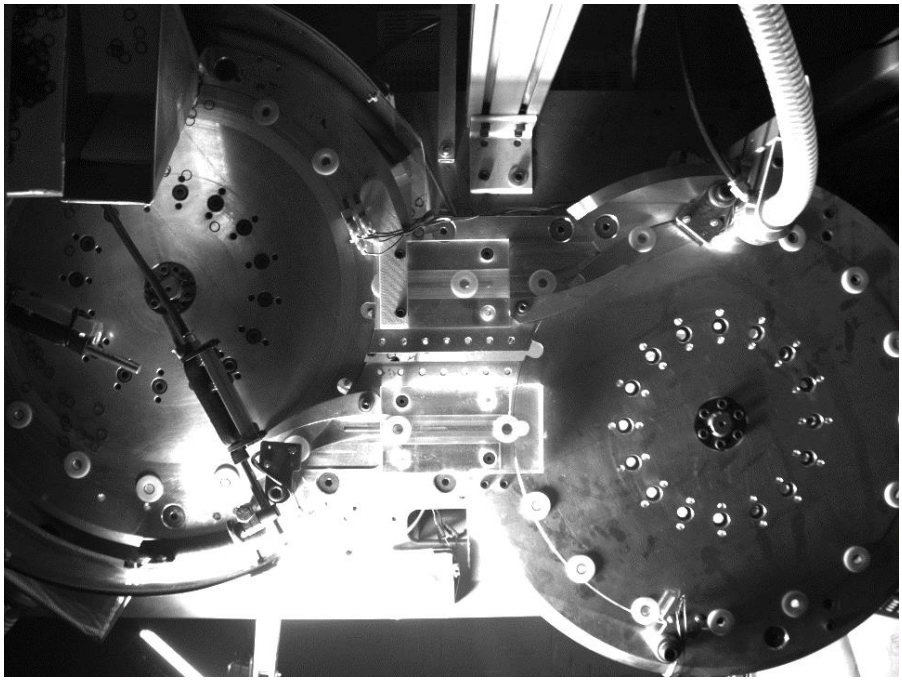


Figure G-2 Only Light 1 was ON

2) Effect of Lights 1 and 2 on image appearance

Figure G-3 shows that Lights 1 and 2 illuminate the entire transfer track ROI (middle region).

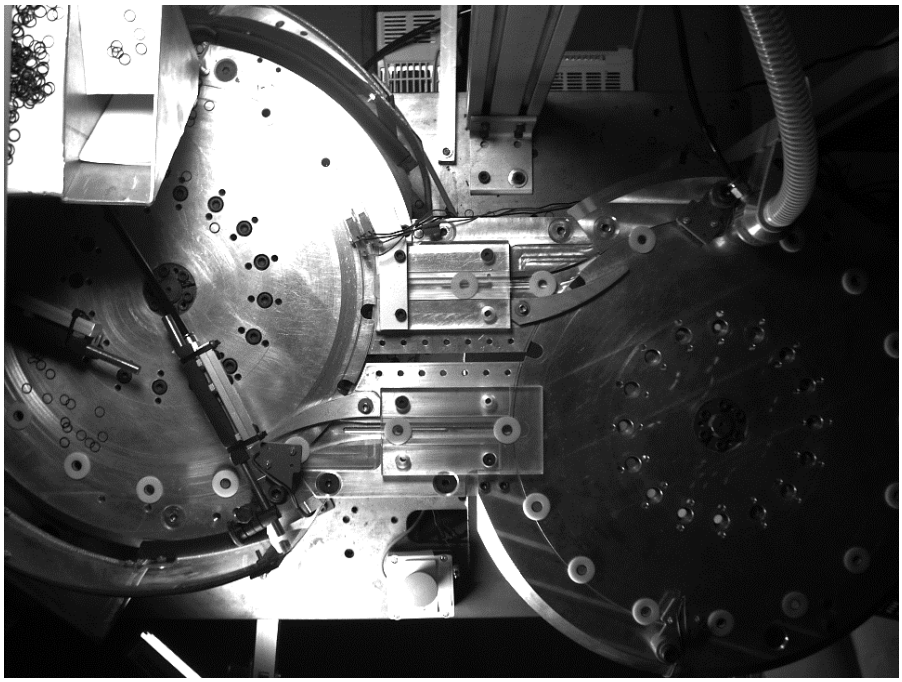


Figure G-3 Lights 1 and 2 were ON

3) Effect of Lights 1, 2 and 3 on image appearance

Figure G-4 shows that Lights 1, 2 and 3 illuminate the transfer track and air knife ROIs.

Figure G-4 Lights 1, 2 and 3 were ON

4) Effect of all four lights on image appearance (without white foam board beneath the wheel)

Figure G-5 shows that all ROIs were properly illuminated with the four lights.

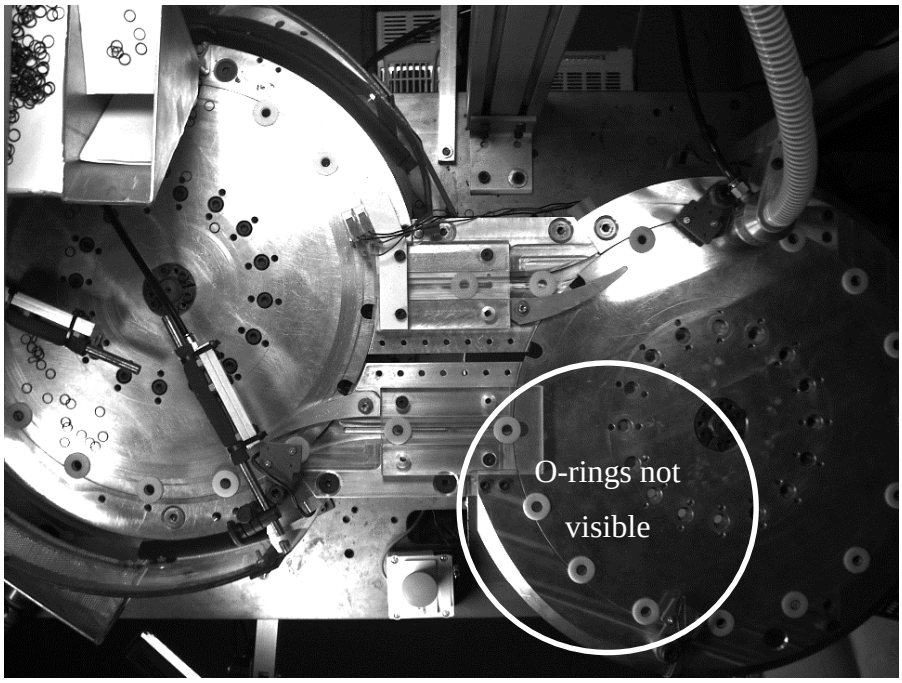


Figure G-5 All lights were ON without foam board beneath the wheel

5) Effect of all four lights on image appearance (with white foam board beneath the wheel)

Figure G-6 shows that all ROIs were properly illuminated with the four lights and O-rings assembled into carriers were also visible on the secondary wheel due to the backlighting effect.

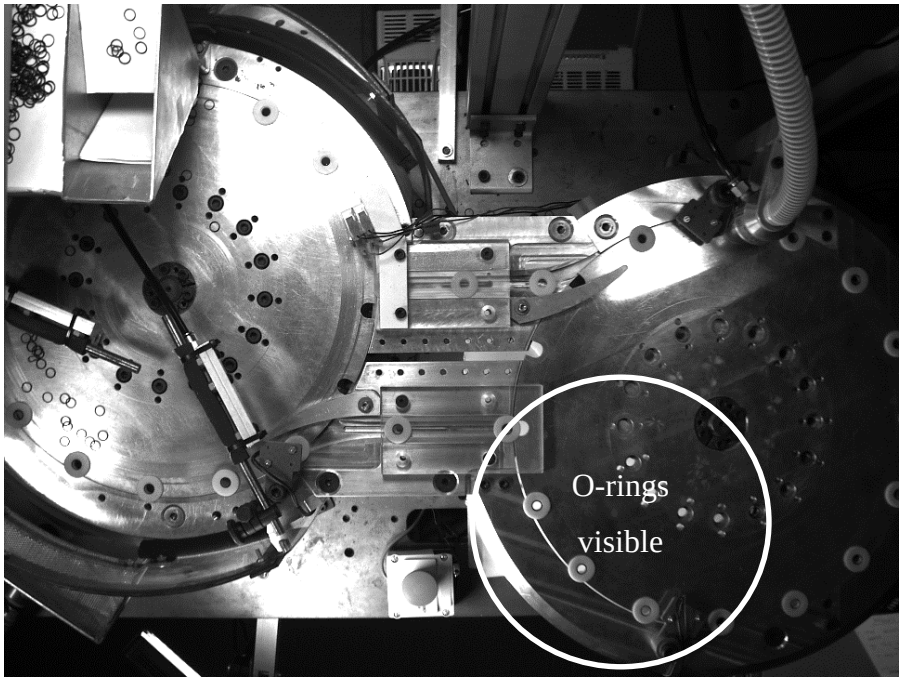


Figure G-6 All lights were ON with white foam board beneath the wheel