Detecting Occlusions of Automobile Parts Being Inspected By a Camera System During Manufacturing Assembly

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Computer Engineering

by
Jayadevan Puthumanappilly
December 2015

Abstract

This thesis considers the problem of detecting occlusions in automobile parts on a moving assembly line in an automotive manufacturing plant. This work builds on the existing "Visual Inspector" (VI) system developed as a joint research project between Clemson University and the BMW Spartanburg manufacturing plant. The goal is to develop a method that can successfully detect occlusions in real-time. VI is a detector and classifier system that uses video cameras to determine the correct installation of a part in the assembly line. In the current version of VI, an occluded part is flagged simply as 'not OK' - as if the part were not installed at all. The new algorithm developed aims to extend the functionality of VI to correctly identify occlusions - i.e., flag an obscured, but correctly-installed part as 'occluded' rather than as 'not OK'. In this thesis, we provide a background of the current VI system deployed at the manufacturing plant. We then discuss the design of an algorithm that recognizes occlusions. Details of tests conducted to verify the correctness of the design, as well as the results of the tests run on real-world data from the plant are presented. Finally, we discuss the possible enhancements to this algorithm as part of future work.

Acknowledgments

I would like to thank my advisor Dr. Hoover, for his help and advice throughout my Master's program. This thesis would not have happened without his patience and guidance. I thank my committee members Dr. Brooks, and Dr. Wang for their time in reviewing this thesis. I also thank Dr. Jörg Schulte from BMW Manufacturing, for acting as the liason for this research project.

My team members Jung Phil Kwon and Ryan Mattfeld were always willing to provide guidance and support - be it explaining the workings of VI, or analyzing code and output. My sincere thanks go out to them. I would like to thank John Hicks, David Moline, and Robert Teague from the ECE Support Shop for their work in setting up the simulation testbed.

Finally, I would like to thank my wife and my parents. Without their support, my Master's degree would have been just a dream.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis considers the problem of correctly identifying occlusions in parts on a moving assembly line in an automotive manufacturing plant. The work contained within this thesis is meant to extend the functionality of the "Visual Inspector" (VI) system developed as a joint research project between Clemson University and the BMW Spartanburg manufacturing plant. VI is a camera-based system that monitors the correct installation of parts during assembly [1]. An occlusion occurs whenever the line-of-sight between the camera and the object being monitored is obstructed for a brief period of time. The length of this time period depends greatly on the quality of image captured by the video camera, which in turn can depend on a number of factors, such as the rate of movement of the part on the assembly line, the width of the inspection area defined for that particular installation of VI, the and the quality of the match between the image of the part captured and the template of the image stored internally within VI. An occlusion can also happen if there is insufficient light on the object being inspected, like when a worker at the assembly line steps in front of the light and casts a shadow on a part being inspected by VI. This shadow can negatively affect the quality of the match that VI calculates.

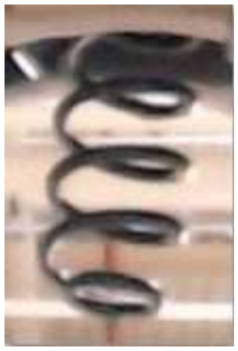

Figure 1.1: Template image of a correctly-installed spring.

1.1 Background of VI and the problem of occlusions

VI is intended to operate on an assembly line, inspecting each car that passes in front of its camera. The two basic classes of problems handled within VI are detection, and classification [1]. The goal with detection problems is to determine whether a part has been installed or not. This is accomplished by storing a series of image templates of the part being monitored within VI, and comparing each of these templates with the live image of the part captured by the camera. Template matching is then performed to determine the level of 'match' between the two images. This type of problem is simpler to set up and operate, and does not require VI to be connected to the BMW vehicle database. Figures 1.1 and 1.2 show template images of an installed spring and a missing spring, respectively. The two template images are used during the detection process to determine whether the cars in figures 1.3 and 1.4 have the springs properly installed or not.

With classification problems however, VI determines whether the correct part has been installed from a set of two or more parts. The reason behind this 'variability' in parts is because every car manufactured by BMW can be customized according to customer specifications. The customer can choose, for example, the type of engine to be installed. Depending on the engine type, different types of badges stating the vehicle model may be applied on the side of the front doors. There are a plethora of such options that are available to the customer, and these options also vary according to the particular vehicle

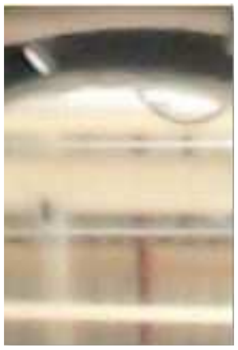

Figure 1.2: Template image of a missing spring.

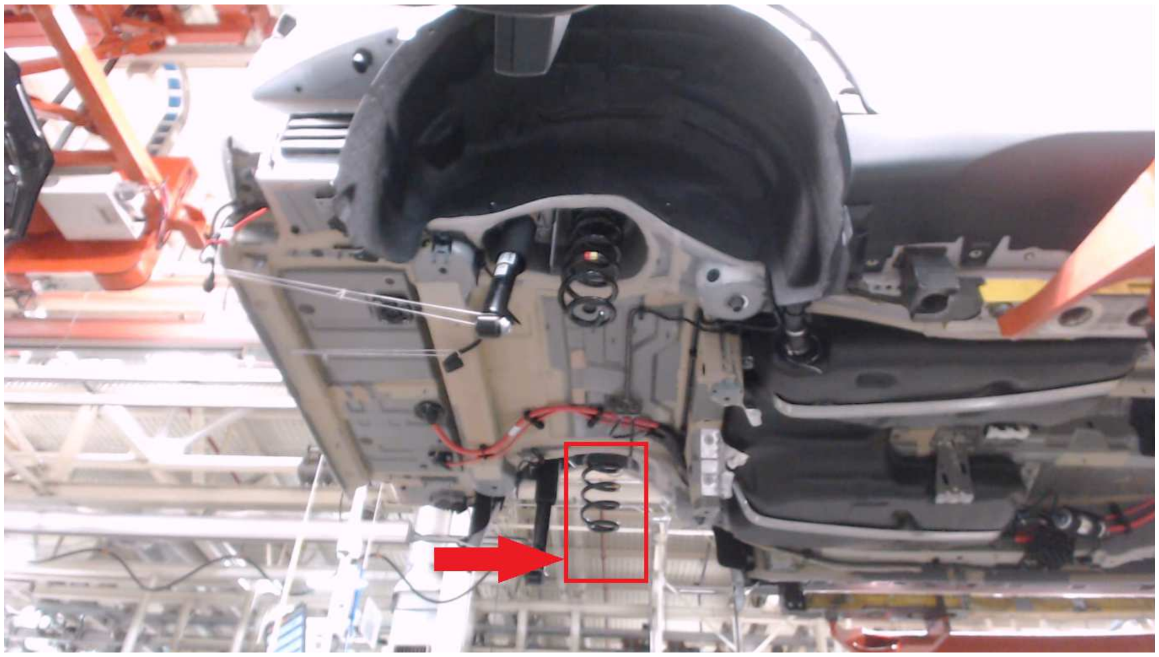

Figure 1.3: Correctly installed spring on the underbody of a car.

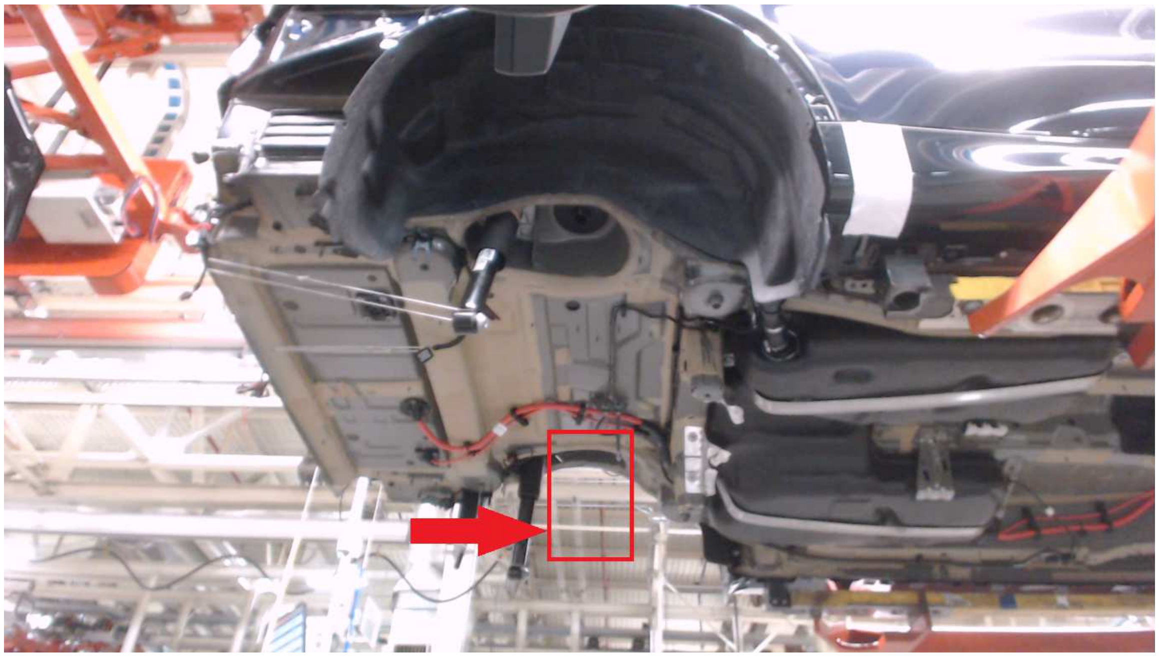

Figure 1.4: Missing spring on the underbody of a car.

model. Thus, no two cars are built the same. To aid in classification, VI connects to the BMW vehicle database to retrieve information that specifies what part is applied to a particular car. Figures 1.5 and 1.6 demonstrate two (of many) possible badges that may be applied on the door-line. For such classification problems, VI needs to connect to the vehicle database in order to verify that the part it detects is indeed the correct part that should be applied on the vehicle that is being inspected.

Occlusions happen whenever the line-of-sight between the camera and the part to be inspected is blocked. This problem occurs primarily because of the nature of work in an assembly line: workers on the shop floor are constantly mobile, installing and inspecting various parts on the assembly while it is in motion. Therefore, it is not uncommon for a worker to accidentally step in front of the VI camera while it is attempting detection/ classification. For any inspection, if the camera is blocked for more than a specific number of image frames, VI flags the occluded part as though it were not installed. Manual intervention is then required by the human operator to acknowledge the error and to let VI proceed

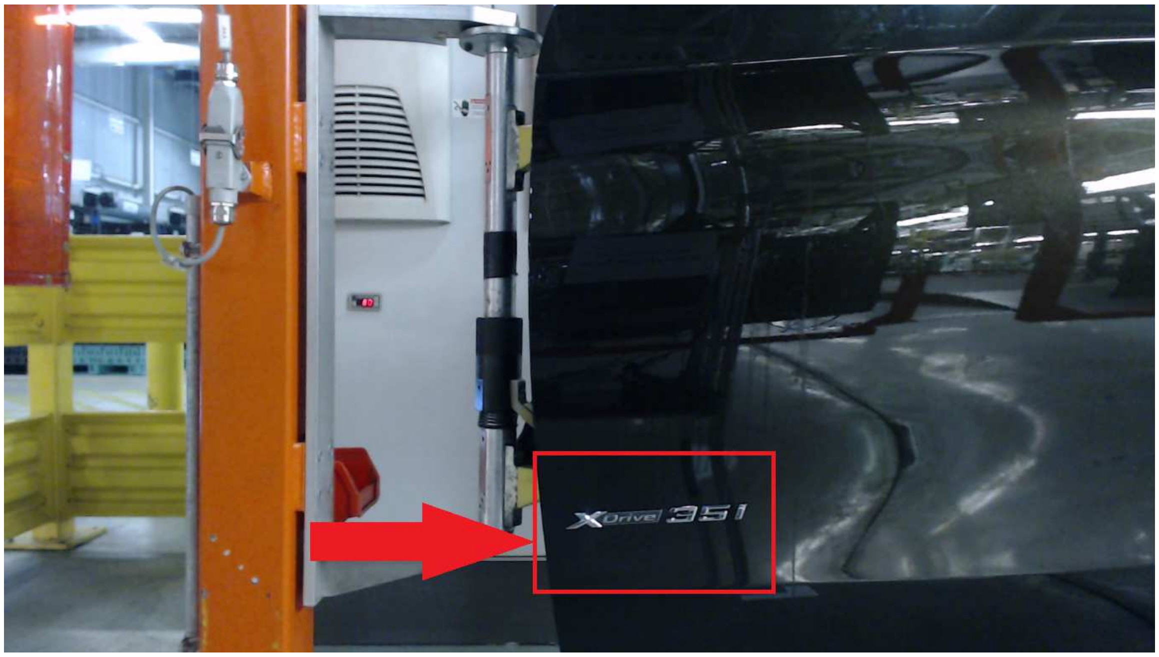

Figure 1.5: Classification of emblem on the door-line for car#1.

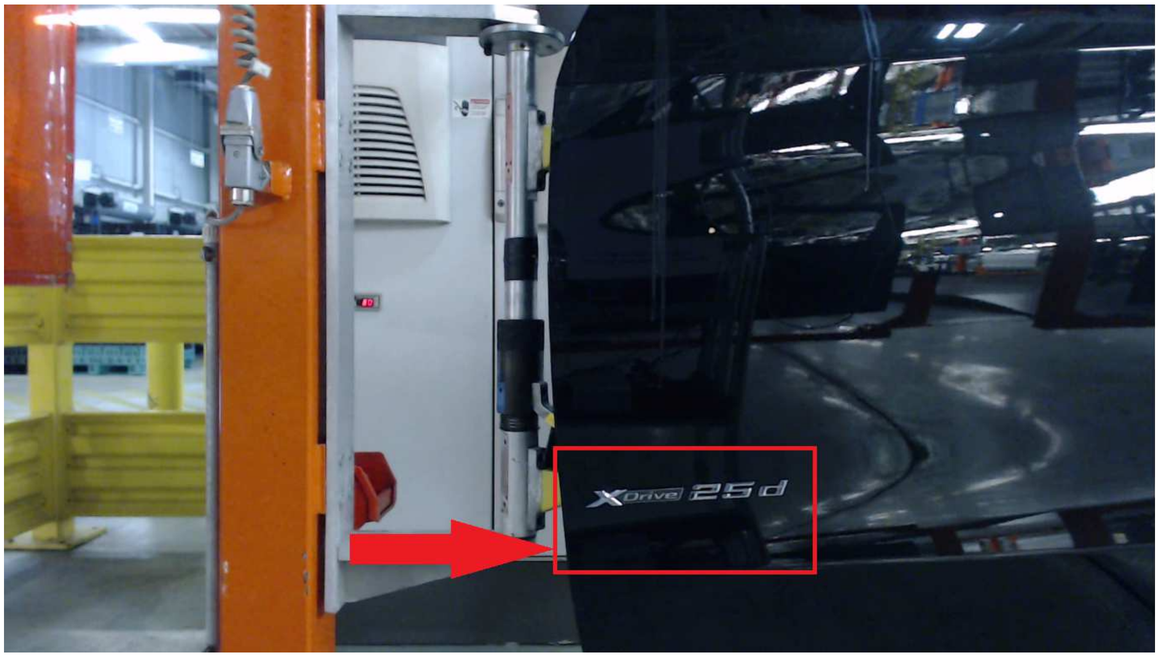

Figure 1.6: Classification of emblem on the door-line for car#2.

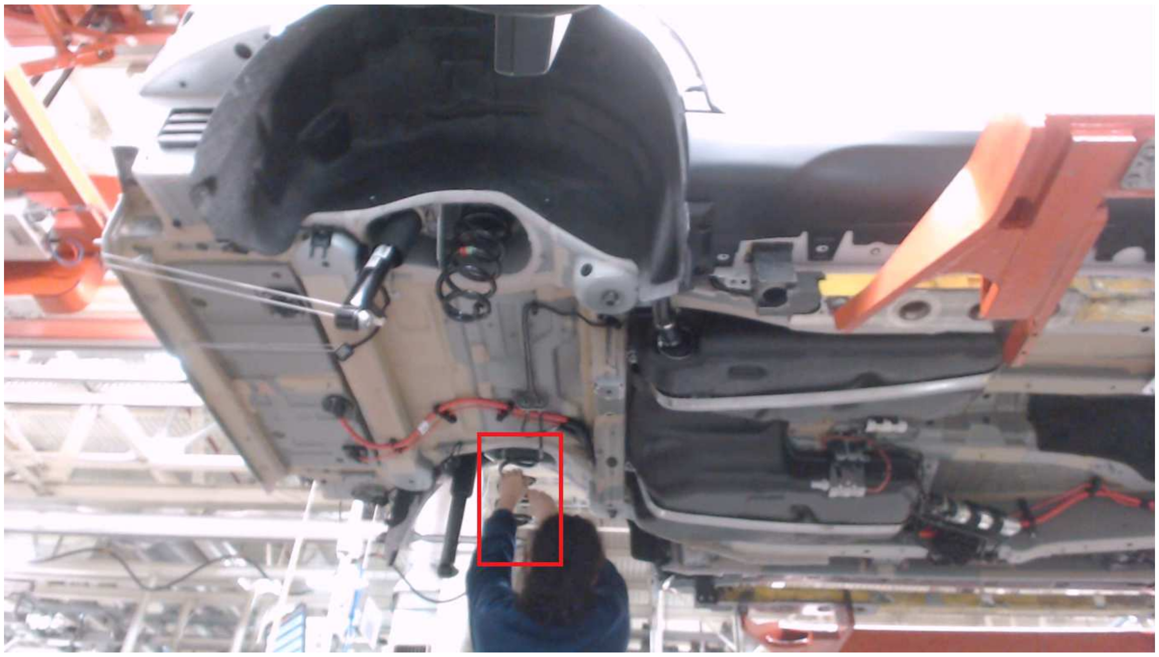

Figure 1.7: Occlusion of the spring during inspection of the underbody.

with the inspection of the next car in line. Figure 1.7 shows an occlusion happening during inspection of the underbody. The worker is making adjustments to the spring installation on the left-rear side of the car while VI is performing its inspection.

1.2 Related work in occlusion detection

The problem of occlusion detection has been studied extensively before. Occlusion detection functionality is usually a part of tracking systems that deal with analyzing information within complex, open-ended scenes. Two areas where occlusions happen frequently are in the fields of vehicle tracking and people tracking. However, both these fields are very different from the relatively simpler problem of tracking objects and detecting occlusions in an assembly line environment. Vehicles being tracked on the roads will display variable speed and sometimes variable directions. The occlusions can happen anywhere within the field of view and lighting conditions and visibility can vary wildly. Figure 1.8 shows the problem of occlusions while tracking vehicles on the road. Tracking people is also a difficult problem to solve. People vary their speed and trajectory constantly. The appearance of a person may change dramatically based on their posture or clothing. Occlusions happen very frequently, and at multiple locations in the field of view. Figures 1.9 and 1.10 show the problem of occlusions while tracking people. The goal of our research is to detect occlusions in a simpler environment within a vehicle assembly plant where the part being inspected is always at a consistent distance from the camera. The motion is predictable, and all the different parts are known beforehand. The system only needs to view the parts that are within a small window, and detect occlusions within that window. The environment is much less complex than one in which a traditional tracking and occlusion detection system is deployed. In this kind of environment, the simple system we have designed may be more robust, and detect occlusions with higher accuracy than complex algorithms that are intented to analyze detailed and intricate scenes, where the tracking problem becomes much more difficult, and occlusions happen very frequently.

The following work in areas of vehicle and people tracking further demonstrate the complexity of the related work. References [5], [7], [8], and [10] involve work that detects occlusions while tracking vehicles on roads. The work described in [5] addresses the problem of occlusions occuring in a vehicle tracking problem, by proposing that the vehicle or object being tracked be split into non-homogeneous segments. These segments are then tracked individually. The idea behind tracking segments, rather than the object as a whole, is that occlusions tend to occur locally within small regions, and that robustness to occlusion can be improved since not all segments are occluded at the same time. The success of this method hinges on the detection of suitable features that can be then used to segment the object into non-homogeneous parts. The work in [8] looks at tracking vehicles from a camera positioned close to the ground, where the height of the vehicles cause occlusion to vehicles behind them. Feature points are selected and tracked through a series of image frames using the Kanade-Lucas-Tomasi feature tracker. The height of the features are also estimated. The features are classified as ‘stable’ and ‘unstable’, depending on the mean and variance of the estimated height. The stable features are used to refine the height of the unstable

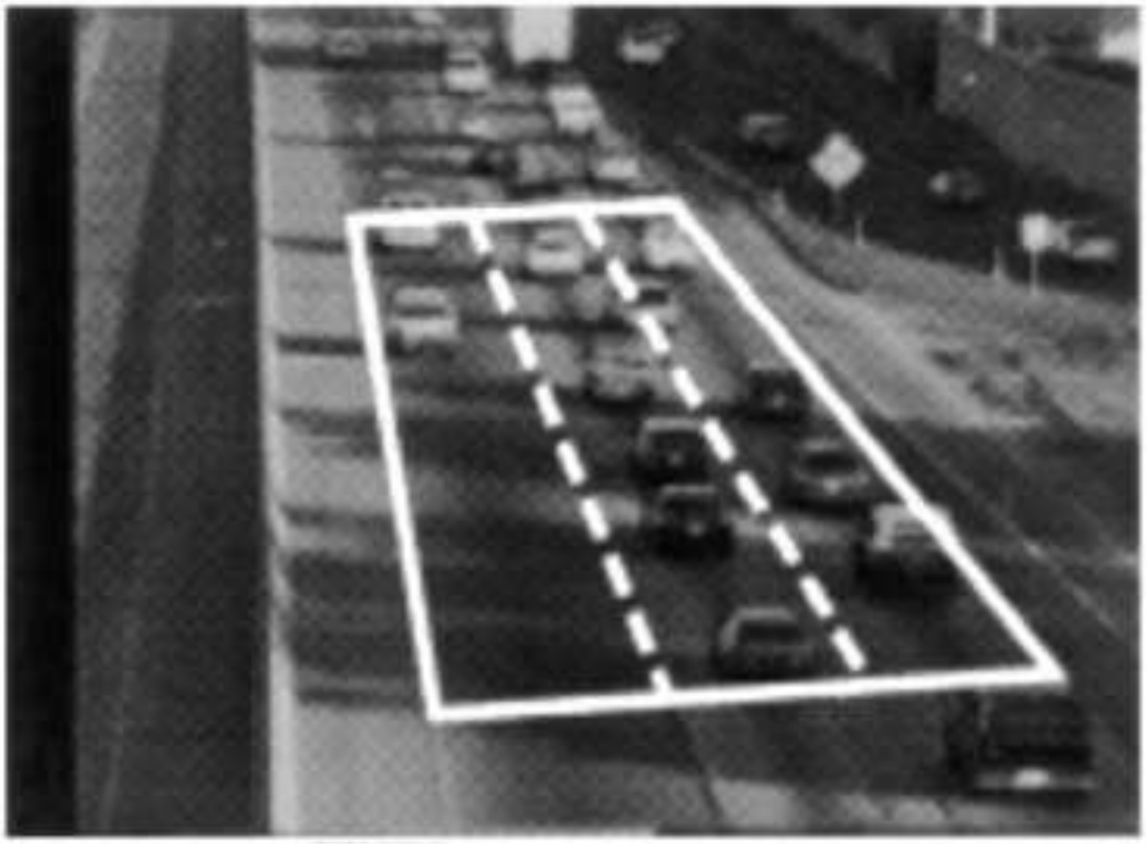

Figure 1.8: Occlusion while tracking vehicles travelling along a highway. Image borrowed from [4].

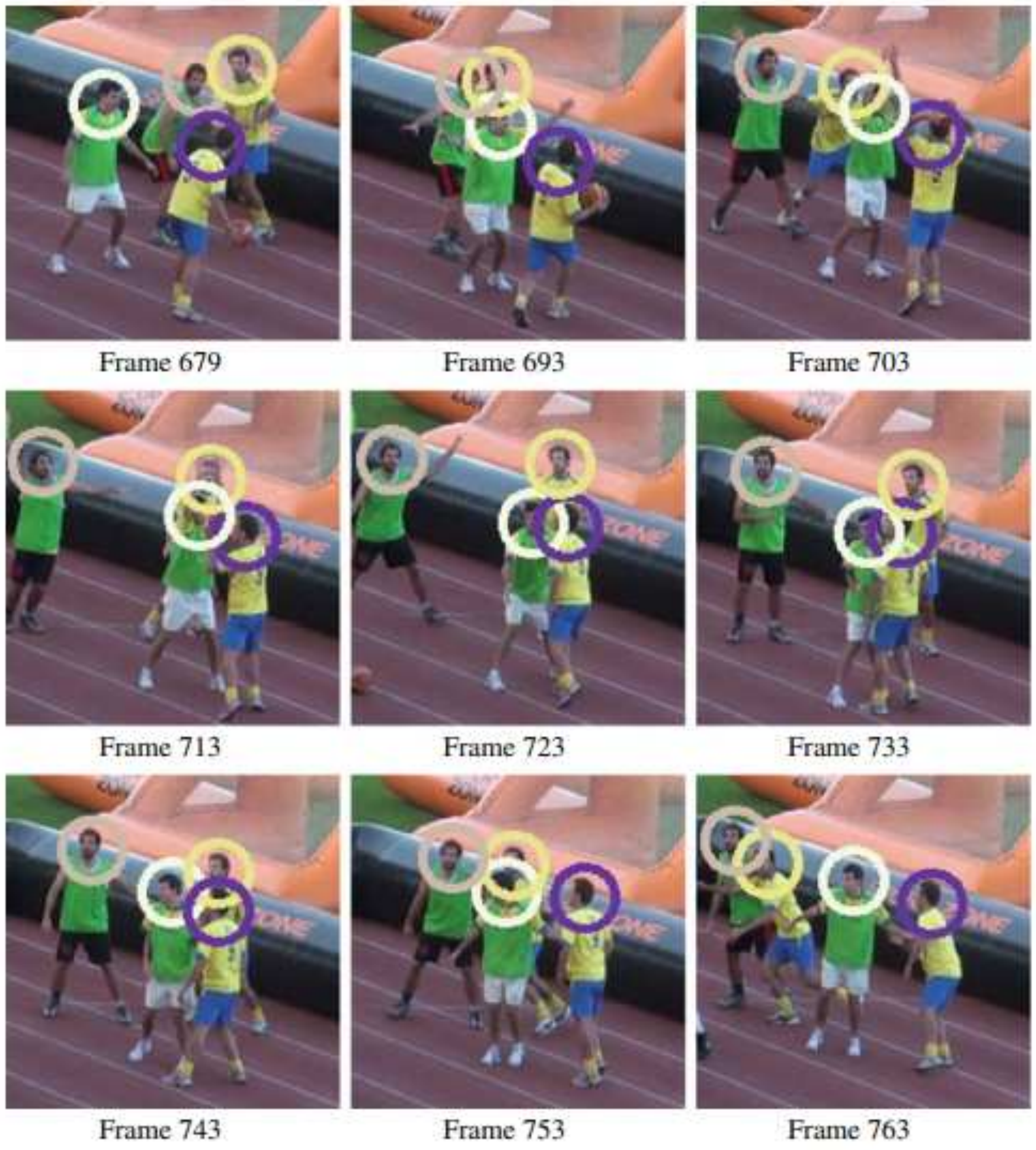

Figure 1.9: Occlusion while tracking people at a track and field event. Image borrowed from [3].

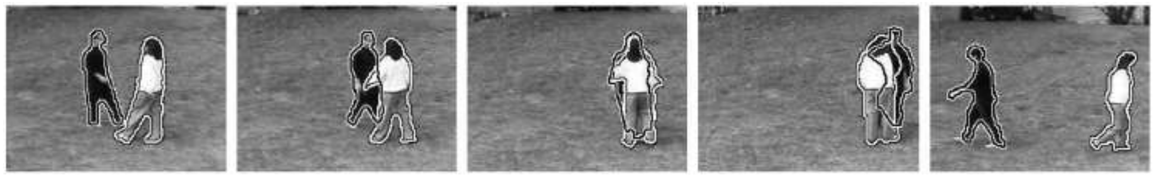

Figure 1.10: Occlusion while tracking two people crossing paths. Note how each persons' silhouette changes during tracking. Image borrowed from [2].

features, and it is this refinement process that makes the proposed system robust when faced with occlusions. The feature points are then grouped together to segment and track the vehicle. The paper in [7] discusses occlusion detection while performing vision-based vehicle identification on highways. The concept of 'motion vector' is introduced, which is a set of values for each vehicle that indicate the rate of movement of specific points on the vehicle. Vehicles are moving in the same direction throughout the field of view of the camera, and hence the motion vector contains homogeneous values for a single vehicle. The process detects occlusions by analysing the variance within the motion vector for vehicles. If the variance is greater than a pre-defined threshold, the motion vector is non-homogeneous, and an occlusion is said to have occurred. The algorithm then attempts to separate the occluded regions by identifying and removing the boundary that overlaps between the two regions. The work in [10] detects occlusions by using apriori information like vehicle-shape models, camera calibration, and groundplane information. The process describes a static camera placed near ground-level that tracks vehicles passing through a nearby roadway. The occlusions in this case can happen from either static occluding objects like trees or poles in the line of sight of the camera, or from dynamic sources (vehicles occluding other vehicles). In the case of static occlusion objects, the process takes the groundplane information into account while analyzing the scene. For dynamic occlusion objects, the algorithm utilizes generic shape models of classes of vehicles to determine when two objects may be occluded.

References [9], [11], and [12] involve work that detects occlusions in scenes of people moving about in real life. Reference [9] describes a visual method to detect and track soccer players on a field. This method detects occlusions that happen between players during the game. Players are first detected by analysing all the pixels across the playground, and determining whether the pixel belongs to an object (a player, or lines on the field) or not. This is done by checking the color intensity difference between the pixel and its immediate neighbours. The idea is that pixels belonging to the background (the playground) will have a uniform color, and hence, will not be part of an object. Players are thus detected as fixed-size regions. Occlusions are detected by tracking the center of gravity of all player regions across frames - if for any two regions, the center of gravities overlap, an occlusion is detected. The work [11] specifies that occlusions happen when pixels around the target being tracked move into the bounding box around the target. The process considers the target object as the foreground, and other regions as the background. An occlusion is detected whenever the background around the foreground appears within the target's bounding box. A Multiple Instance Learning method and Support Vector Machine is used to train the classifier that detects foreground and background objects. The study is made on subjects moving down a corridor in a building, so that depth of movement comes into consideration. The work within [12] utilizes depth information to detect occlusions as they happen. The process uses an efficient belief propagation algorithm [6] to generate dense depth images from binocular video sequences. A bounding box is assigned to the person being tracked. To detect occlusion, the process assumes that the median depth value of all pixels in the bounding box (the dominant depth value) changes when an occlusion happens. If the change in the dominant depth value over two images frames is greater than a threshold, an occlusion is detected.

1.3 Novelty

VI was designed to be very low-cost, and easy to set up and operate. It uses inexpensive cameras and computers to perform inspection in real-time, and its detection and classification algorithms have been developed in-line with these limitations. Algorithms employing expensive image computations were ruled out, and the focus was on developing a simple algorithm that could use image statistics to detect occlusions. To our knowledge, such an approach has not been employed previously.

Chapter 2

Research Design and Methods

2.1 Overview of the current inspection process in VI

For the purpose of this discussion, and the rest of the document, the term “car” will be used interchangeably to refer to any assembly (underbody, car door, etc. based on context) that is to be inspected. VI operates at 10 Hz, continuously inspecting 10 images per second. Figures 2.1, 2.2, and 2.3 are a sequence of images recorded as a car door appears for inspection. VI uses a “trigger” to identify when a part is ready to be inspected. The trigger is separate from the part being analyzed - it has to have a consistent appearance and always be visible across all cars during inspection. The trigger is the first step in the inspection process. The “trigger template” specifies the shape of the trigger, and the “trigger search window” specifies an area over the image where the trigger template is searched for. This window is larger than the template, and allows for some degree of variability in the position of the trigger, thus making the inspection process more robust. Each trigger window must have at least one trigger template defined, and more number of templates may be specified if required, to take into account slight variations in the appearance of the trigger. Multiple triggers may be used in case all the parts to be inspected do not fit into a single image frame, and VI currently supports up to four separate triggers per car. VI continuously searches the trigger search window for an appearance of the trigger template. When a trigger template is identified, VI then searches for “inspection templates” within one or more “inspection search windows”. VI supports up to 100 different parts (or “classes”) that can appear within an inspection search window. Classes specify what shapes can be expected within the inspection window - for example, the doorline-emblem use case can have any one of seven different emblems appearing within the search window, depending on the car model being inspected. Each emblem is defined as a separate class. Each individual class also has one or more “class templates” defined to account for slight variations in appearance. Similar to trigger templates, more number of class templates can be specified to improve the robustness of classification. The inspection search window is larger than the class template to allow for variability in the placement of the class to be inspected. Figures 2.4 and 2.5 demonstrate the concept of trigger template and trigger search window, as well as class template and inspection search window. Finally, VI inspects multiple sequential images taken as the part passes through the inspection search window. This again improves the robustness of classification, and ensures that anomalies occuring in one or more image frames do not throw off the classification.

VI is built using C++, and uses the OpenCV library to perform the template matching process. For simplicity, consider an installation of VI with a single trigger search window and a single inspection search window, with two or more classes associated with the inspection window. For each image frame that it captures, VI inspects the trigger search window for occurrences of the trigger template. The template match process generates a correlation value between 0.0 and 1.0 for all pixels across the trigger search window. A successful match occurs at one or more locations within the trigger search window. The location of the pixel with the highest correlation value within the search window is recorded and this pixel location is used to define the center of a rectangle corresponding to the trigger template. The trigger template is deemed to be located within this rectangle. Once VI locates a trigger template, it starts collecting image frames for the entire duration of time the trigger template is visible within the trigger search window. After the trigger template moves out of the trigger search window, VI sets about classifying all the image

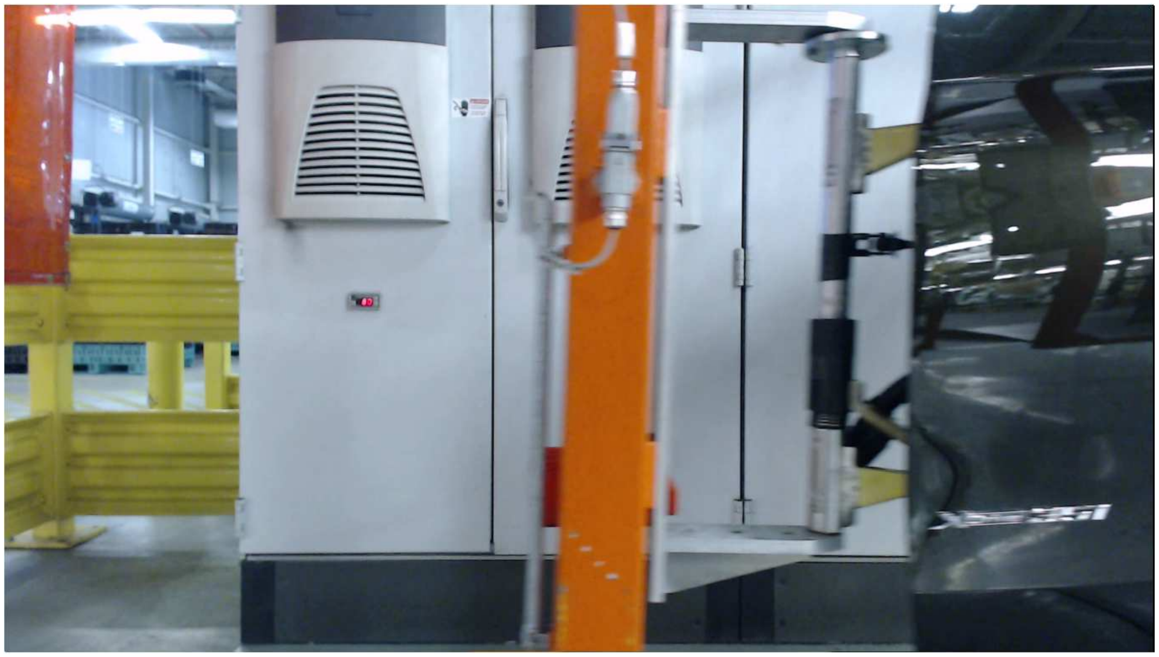

Figure 2.1: Sample image of a car door during inspection, passing from right to left in front of VI.

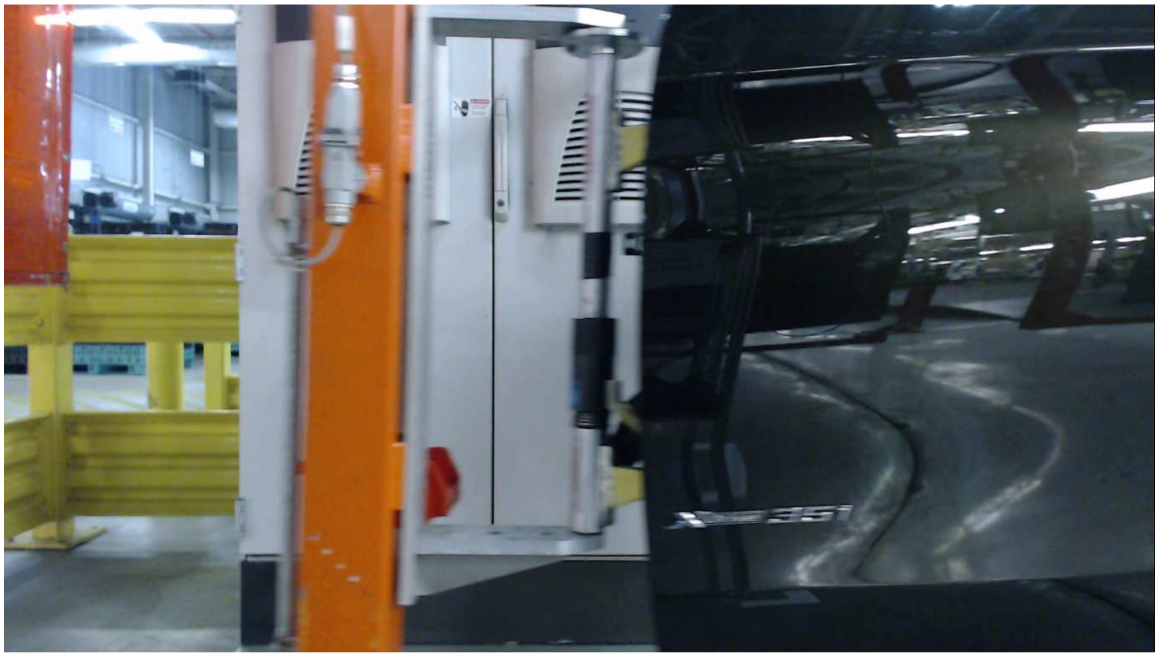

Figure 2.2: Sample image of a car door during inspection, passing from right to left in front of VI.

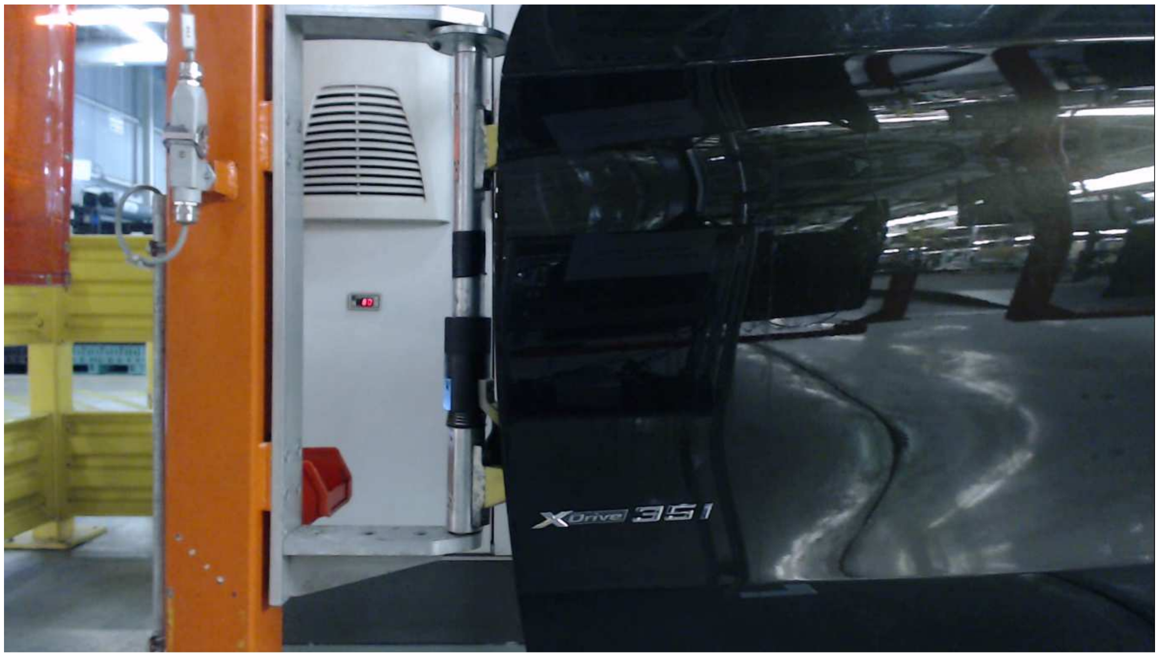

Figure 2.3: Sample image of a car door during inspection, passing from right to left in front of VI.

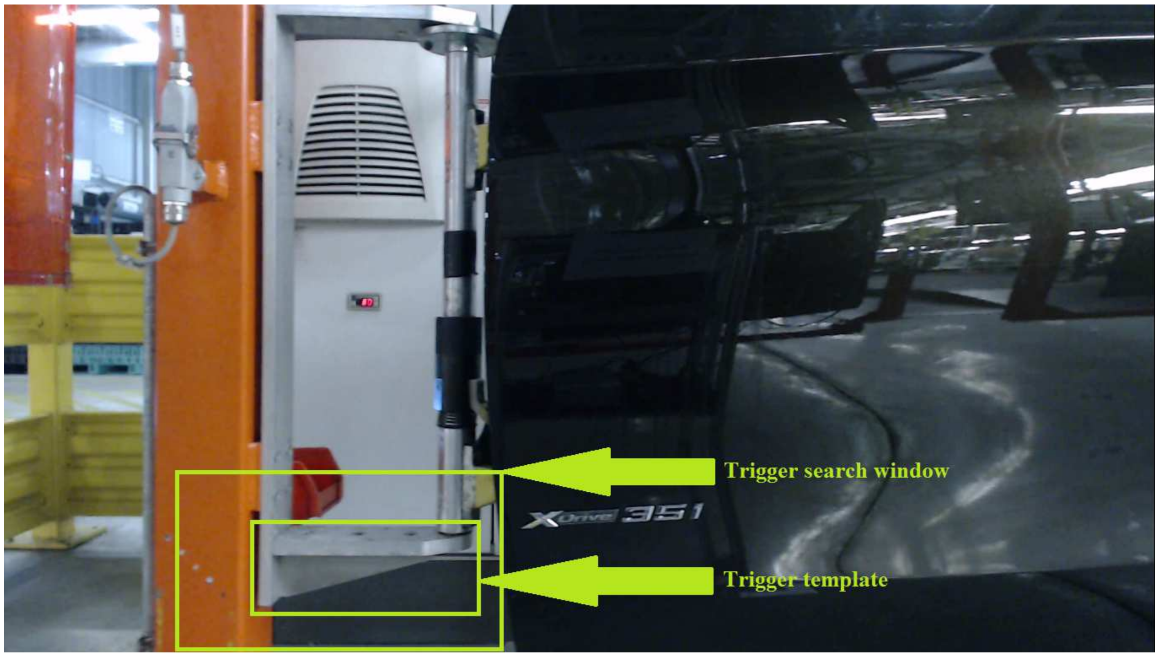

Figure 2.4: Sample image of a trigger search window and template.

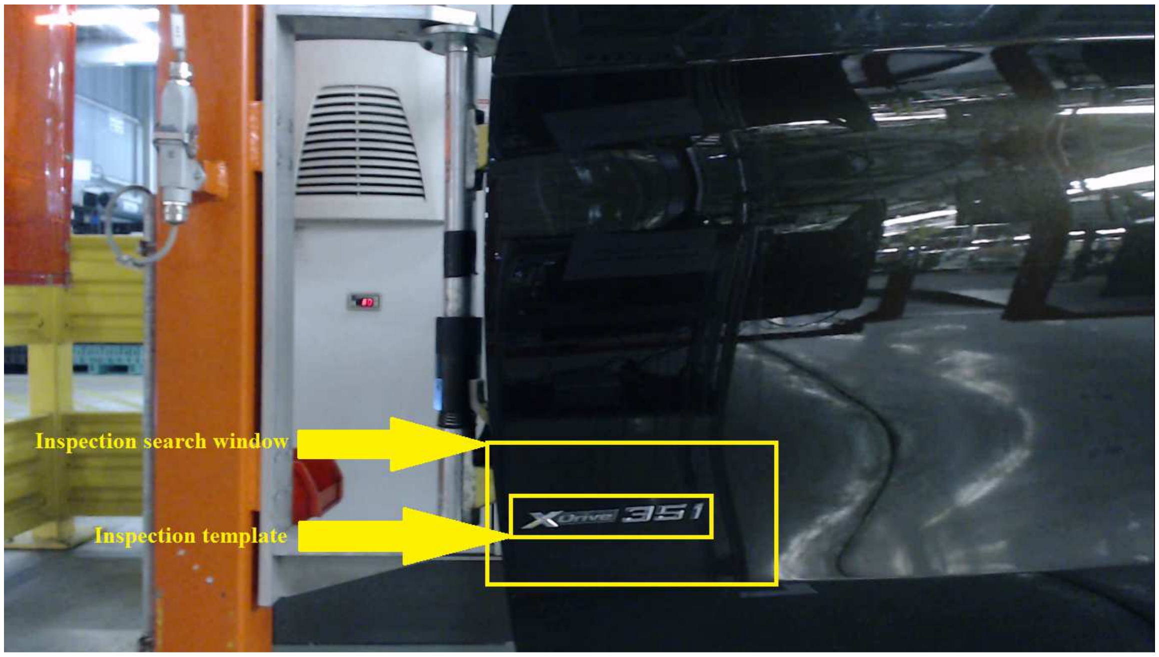

Figure 2.5: Sample image of the inspection search window and template corresponding to the trigger search window and template in figure 2.4.

frames it previously collected. For each image frame, VI scans the inspection search window for the presence of any of the defined inspection templates (corresponding to each of the defined classes for the installation). Template matching is again used in a manner similar to that of the trigger template search process. Each class is assigned a correlation value by the template matching process, and the class with the highest correlation value is voted the 'winner'. The result of this inspection is then displayed on-screen, and the winning class name along with its correlation value is written to a text file.

2.2 Updates made to the existing VI inspection process

The functionality of the current VI inspection process was extended to output additional information which could then be utilized to identify occlusions. Currently, VI calculates the correlation value separately for each class within a given inspection area, but only considers the maximum correlation value across all classes during classification. VI

Table 2.1: The record format of the new output text file from VI.

Field Name	Description
Trigger Frame Index	The frame number of the image frame captured when VI locates a trigger template.
Trigger Value	The maximum correlation value of the trigger template within the trigger search window for this image frame.
Class Frame Index	The frame number of the image frame undergoing classification. This value will match the Trigger Frame Index when VI is not actively classifying image frames.
Inspection Area Number	The inspection area corresponding to each class being considered during classification. This value will be zero when VI is not actively classifying image frames.
Class Correlation Value	The correlation value of the classes undergoing classification within VI. There can be more than one Class Correlation Value field in this file, corresponding to the number of classes defined within this installation of VI. Value will be zero when VI is not actively classifying image frames, or for classes outside of a particular inspection area.

was modified to generate an additional output text file that contains the correlation value of the trigger and all classes within an inspection area, for each image frame captured. The record format of this text file is as per table 2.1.

2.3 Simulation testbed

A testbed was constructed to mimic the movement of cars on the assembly line at BMW. The testbed consisted of a platform mounted on metal rails, a spool attached to an electric motor, and a length of wire from the spool that could clip-on to the edge of the platform. Objects to be inspected were placed on top of the movable platform. When switched on, this setup could move the platform laterally across the field of view of a video camera at a consistent rate of speed. See figure 2.6 for a picture of this installation. Objects on the platform could thus be inspected via VI, similar to how cars are inspected in the manufacturing plant (see figure 2.7). Three classes of objects (or parts) were considered for

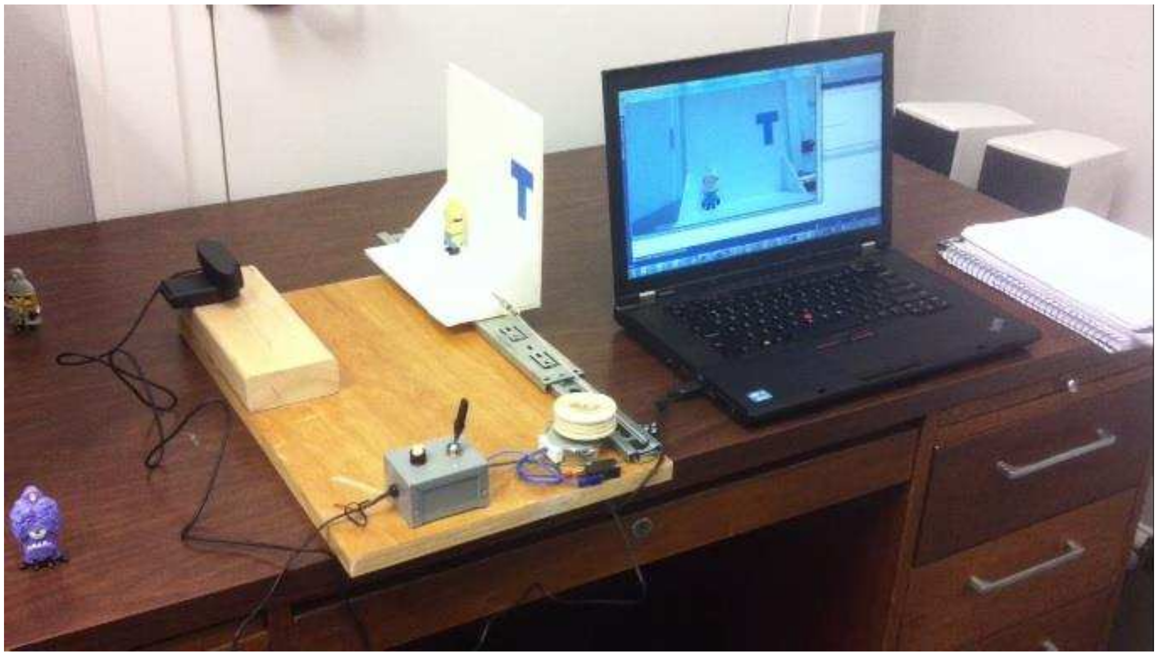
Figure 2.6: Picture of the testbed installation.

this installation, with one template per class. The parts were three 'Minion' characters from the *Despicable Me* movie franchise (see figure 2.8). The three Minion toys were choosen for their similarity in dimensions, to replicate parts inspected by VI at the BMW plant. The rights to Minions lie with Universal Studios Inc.

2.4 Simulation runs on the testbed

With the testbed in place, test runs were made that simulated the types of occlusions that VI encounters in the BMW plant. Table 2.2 specifies the different types of occlusions that were considered, and figure 2.9 shows sample images captured of each type of occlusion, using Class2 as the reference for inspection. Brief occlusions are quick, one-off occlusions that happen once during the inspection. This type of occlusion typically occurs when a worker passes by in front of the part during inspection. The occlusion is only momentary, and VI sees the complete part for most of the inspection process. This was simulated on the testbed by momentarily placing a cardboard box in front of the Minion during

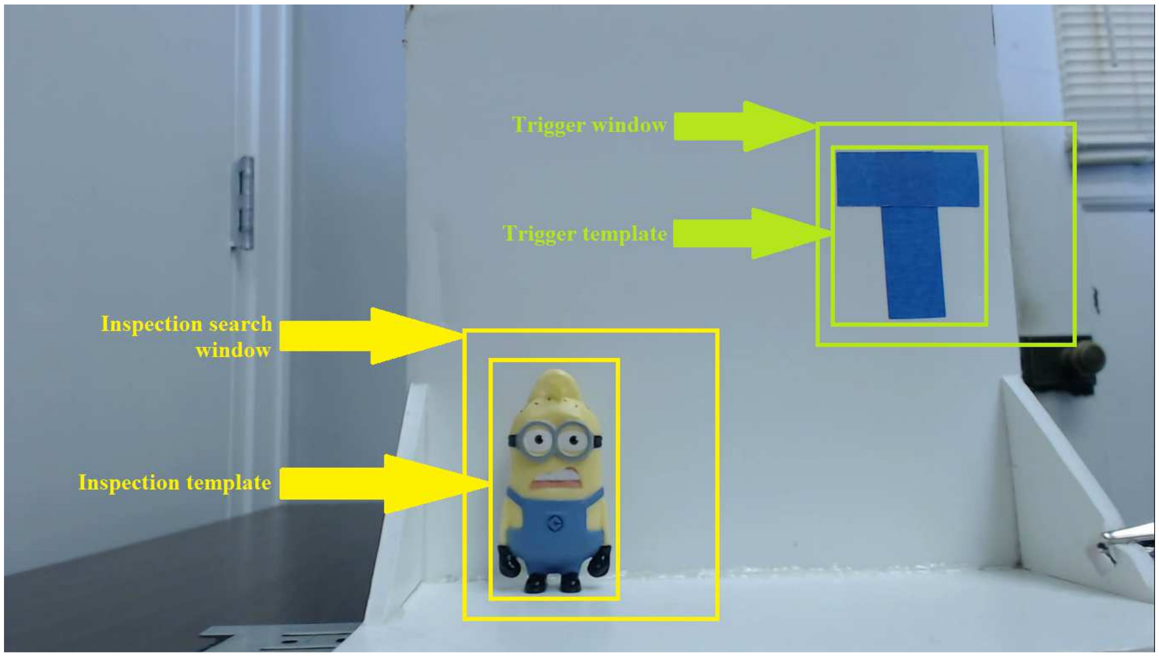

Figure 2.7: Inspection setup for the testbed installation.

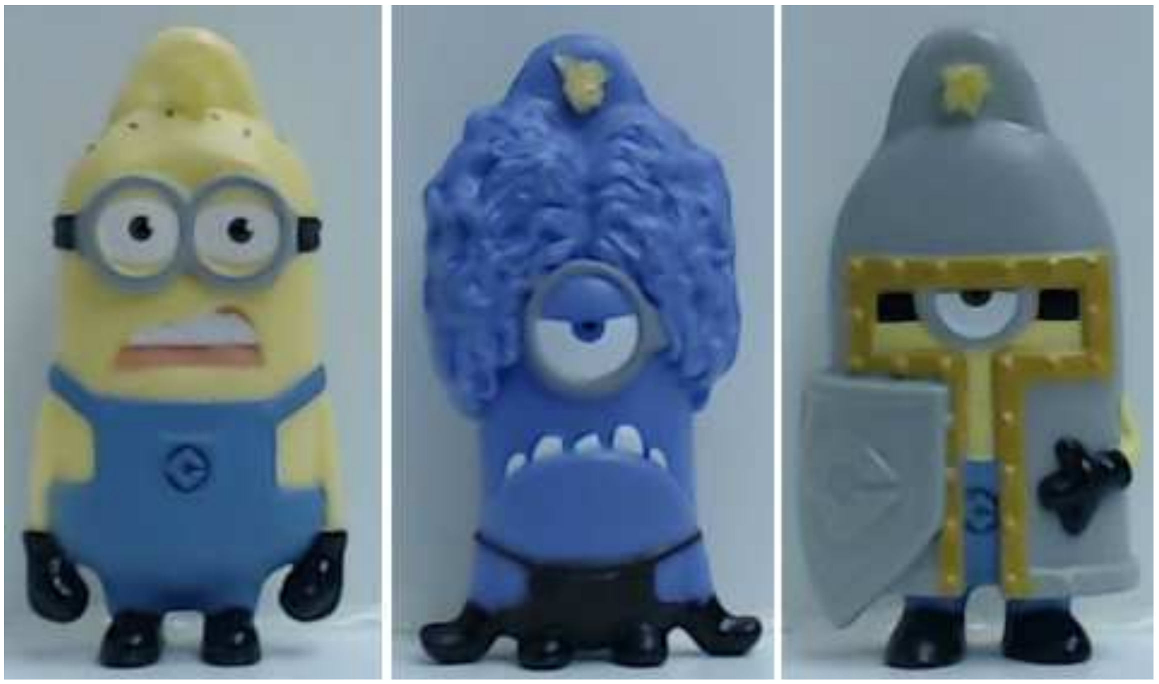

Figure 2.8: The three inspection classes (Class1, Class2, and Class3, respectively) used in the testbed installation.

Table 2.2: The different types of occlusions defined for the simulation.

Type of occlusion	Description
None	No obstruction of the part during inspection.
Brief	A quick, momentary obstruction of the part. The part is fully visible for the majority of the duration of inspection.
Partial	A section of the part is completely obscured for the entire duration of the inspection.
Intermittent	A series of brief occlusions of the part. The part is fully visible in between the sequence of obstructions.

inspection (top right image in figure 2.9). Partial occlusions happen when the part being inspected is partially or completely covered. This type of occlusion typically happens in the plant during assembly, when mutilation tape is not completely removed from the part. The mutilation tape is used to cover a part during assembly to protect it from damage. Sometimes, the worker performing the assembly does not completely remove the tape from the part before the car leaves the assembly location, leaving the part partially obscured. This was simulated on the testbed by applying a piece of electrical tape partially over the Minion during inspection (bottom right image in figure 2.9). Intermittent occlusions are a series of brief occlusions that happen over the course of a particular inspection. This type of occlusion occurs when a worker is actively working on the car currently being inspected, so that his/ her back and forth movements across the part being inspected causes repeated brief occlusions. This was simulated on the testbed by repeatedly blocking the Minion with a cardboard box while it was being inspected (bottom left image in figure 2.9; note that for the intermittent occlusion, although only a single image is presented, it is in effect a series of brief occlusions occuring across multiple image frames. The additional image frames have not been presented here). For all the types of occlusion considered, the common factor is that the trigger is never occluded. VI does not perform classification when the trigger is occluded, and hence, does not generate the correlation information that is vital to our analysis. In all cases in the testbed simulation, care was taken to make sure the trigger was never occluded.

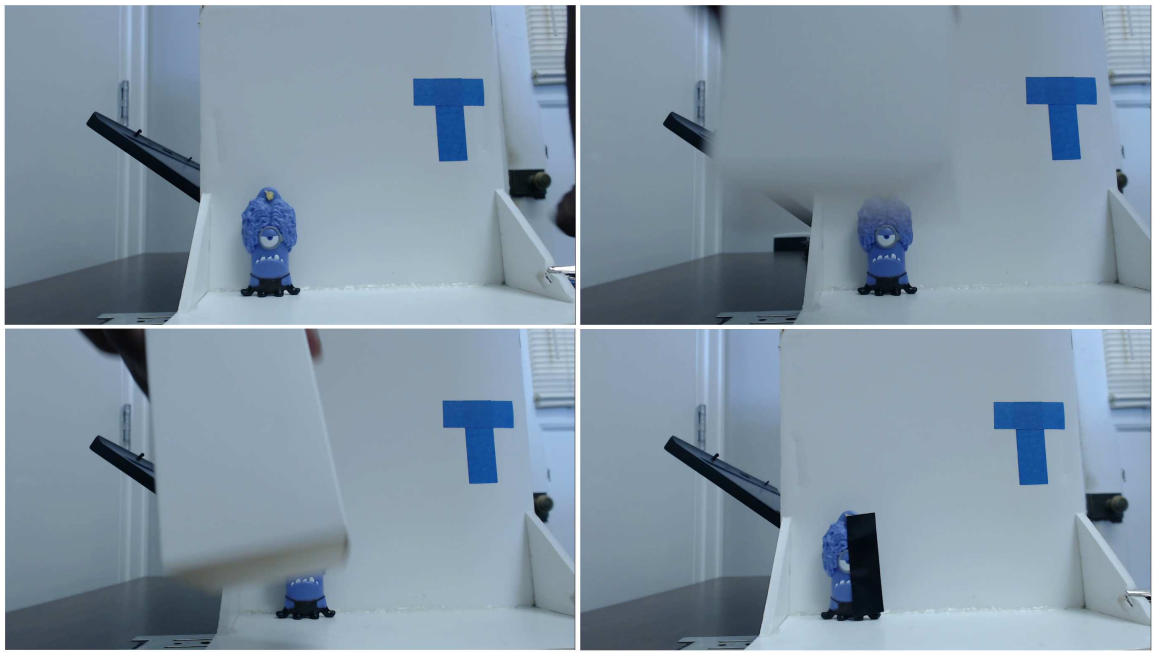
Figure 2.9: Sample images of the different types of occlusions during simulation (clockwise, from top left: none, brief, partial, and intermittent).

Using the testbed, correlation values corresponding to each type of occlusion were generated. Four different runs were made on the testbed using Class2 for the inspection. A different occlusion type was simulated on each run, and the results were noted.

Another simulation was also conducted on the testbed, whereby a sequence of 20 consecutive inspections was run, and the results were recorded. The inspections were set up to represent all the different types of occlusions. The ground-truth data corresponding to this test is specified in table 2.3.

2.5 Analysis of correlation values to detect occlusions

Four cases were run on the testbed, using Class2 as the part to be inspected. A different type of occlusion (none/ brief/ partial/ intermittent) was simulated during each case. For each of the four cases, the results were analyzed, and the new correlation values obtained from VI were plotted. Each plot obtained had distinct characteristics.

Table 2.3: The ground-truth data corresponding to the simulation on the testbed.

Minion no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion type
1	1	0	None
2	1	0	None
3	1	0	None
4	1	1	Brief
5	1	1	Brief
6	1	1	Brief
7	1	1	Partial
8	1	1	Partial
9	1	1	Partial
10	1	1	Intermittent
11	1	1	Intermittent
12	1	1	Intermittent
13	1	0	None
14	1	1	Brief
15	1	1	Brief
16	1	1	Intermittent
17	1	1	Intermittent
18	1	1	Intermittent
19	1	1	Brief
20	1	1	Partial

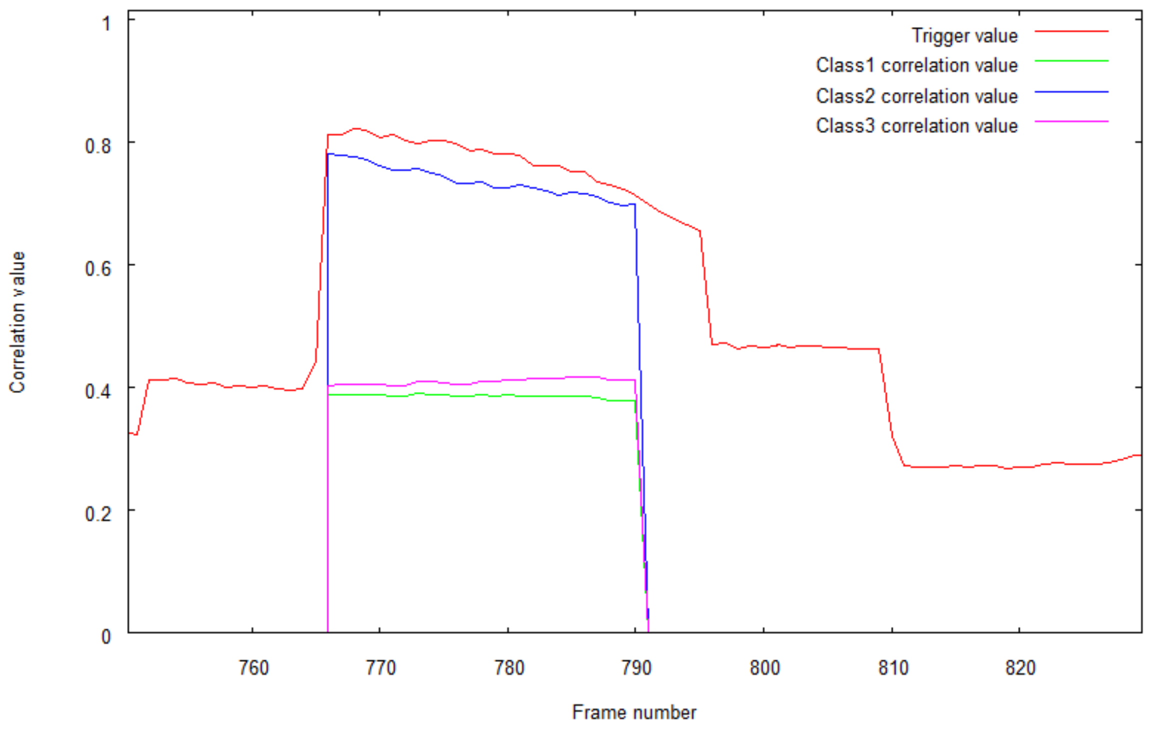

Figure 2.10: Plot of trigger and class correlation values corresponding to each image frame for an inspection with no occlusion.

For an inspection with no occlusion, Class2 correlation values (the light blue line) are high, and the correlation values are more or less stable for the duration of the inspection. This graph is presented in figure 2.10. The correlation values for the other two classes are low (but not zero). The fact that the correlation values for all three classes is stable was noted.

For an inspection with a brief occlusion, class correlation values demonstrate a downward spike for the few image frames that were occluded. Note that the correlation values corresponding to the class being inspected (Class2) demonstrate a much larger fluctuation than the correlation values for the other two classes. Also not that the trigger (the red line) was not occluded during this inspection. All frames that did not experience occlusion had high correlation values. This graph is presented in figure 2.11. The change in correlation values for all three classes across the inspection was noted.

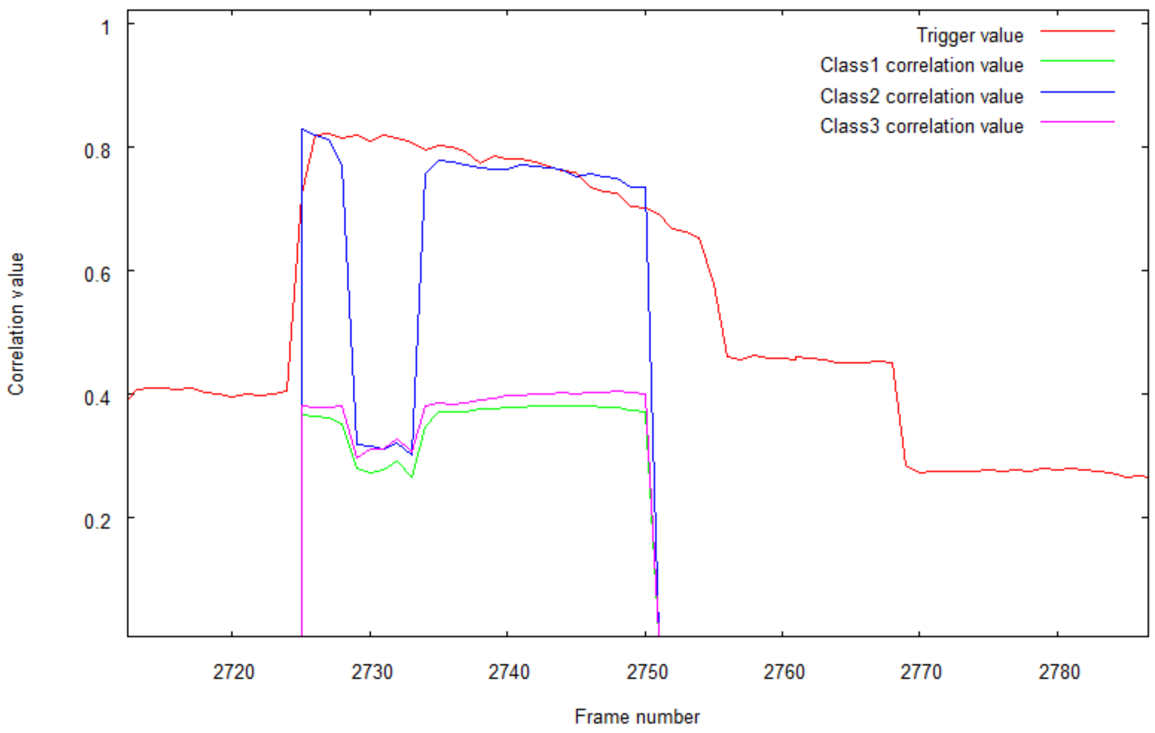

Figure 2.11: Plot of trigger and class correlation values corresponding to each image frame for an inspection with a brief occlusion.

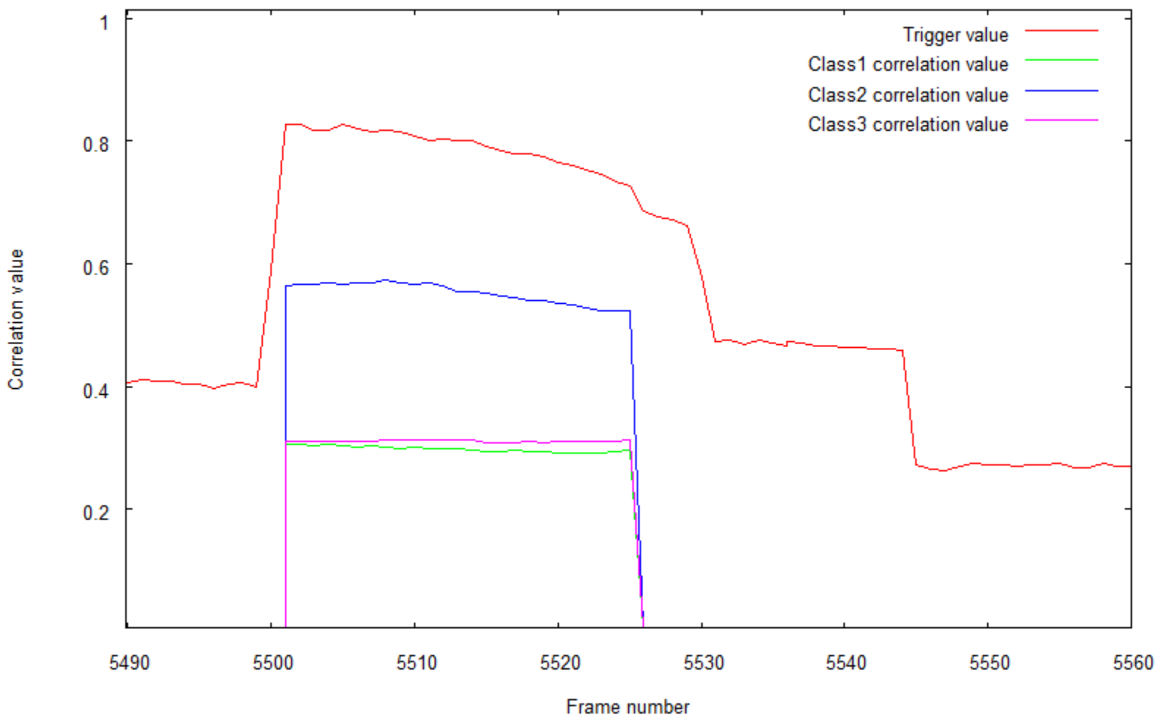

Figure 2.12: Plot of trigger and class correlation values corresponding to each image frame for an inspection with a partial occlusion.

For an inspection with a partial occlusion, the correlation values corresponding to Class2 are very low, but more or less stable. The plot is similar to that of an inspection with no occlusion, but with much lower class correlation values. The correlation values for the other two classes are similar to the values seen for the case with no occlusion. This graph is presented in figure 2.12. The fact that the correlation values for all three classes is stable was noted.

For an inspection with an intermittent occlusion, all the correlation values demonstrate downward spikes during the image frames that were occluded. The trace for Class2 shows the largest fluctuation, whereas the trace for the other two classes show less deviations. The un-obstructed frames had high correlation values for Class2. Note that the trigger was not occluded during this inspection. This graph is presented in figure 2.13. The change in correlation values for all three classes across the inspection was noted.

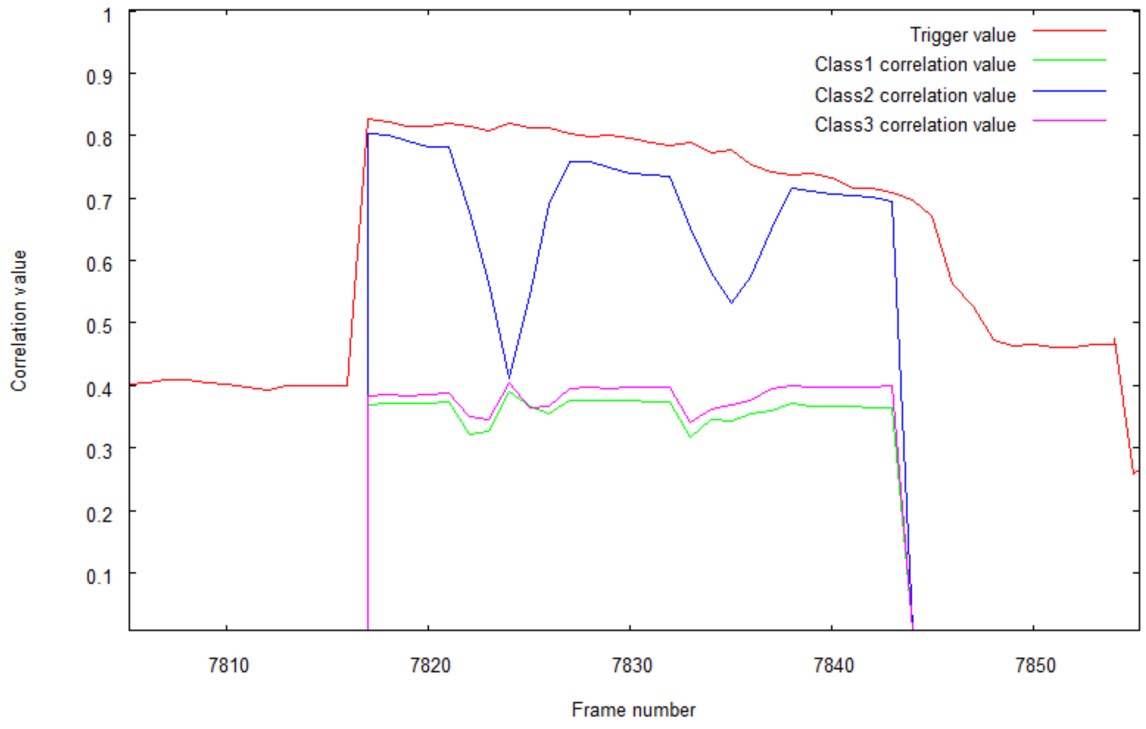

Figure 2.13: Plot of trigger and class correlation values corresponding to each image frame for an inspection with intermittent occlusions.

It is evident from the plots that brief and intermittent occlusions result in a large variation in the class correlation values. Inspections having partial occlusions do not demonstrate this variation. Nevertheless, we suggest that the average standard deviation of the class correlation values within each inspection area is a good indicator of whether an occlusion has occurred - if a class demonstrates a high value of standard deviation in correlation values, it is reasonable to infer that a brief or intermittent occlusion has happened during inspection. However, if a class demonstrates low standard deviation, it could mean that there was either no occlusion at all, or that there was a partial occlusion - the proposed new algorithm does not recognize partial occlusions. The algorithm will track the correlation values for the trigger and each class independently for each inspection, and utilize the standard deviation values to identify brief and intermittent occlusions.

2.6 Details of the new algorithm to detect occlusions

2.6.1 Statistics used to detect occlusions

For each inspection area to be analyzed the algorithm requires as input, the correlation values generated for each class within that inspection area. The class correlation values are generated for the entire duration of the trigger (the time between when the trigger correlation value first rises above the trigger threshold value, and when the trigger correlation value falls below the threshold for more than 10 frames). The class correlation values can be obtained from VI, from its existing processing. For each inspection area, define the set of correlation values as:

$$X_C = x_1, x_2, x_3, ...x_N, \forall C = 1 \text{ to } M,$$

where N is the duration of the trigger in frames,

and M is the number of classes within this inspection area

The algorithm also needs the threshold value for standard deviation, σ_U as input. This is a single value for each use case U. For the details on how this value can be obtained, refer to section 3.2.

For each class C in the inspection area to be analyzed, compute the standard deviation of the correlation values X_C as:

$$\sigma_C = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2} \tag{2.1}$$

After computing each σ_C, calculate the average standard deviation of correlation values across all the classes in the inspection area as:

$$\sigma_I = \frac{\sum\limits_{C=1}^{M} \sigma_C}{M} \tag{2.2}$$

Finally, if $\sigma_I > \sigma_U$, the current inspection area has an occlusion, and this is to be flagged.

2.6.2 Context of the new occlusion detection process with respect to existing VI processing

The new algorithm has been developed as a stand-alone application for the purpose of this research, but the ultimate aim is to integrate it with the existing detection and classification process within VI seamlessly, so that VI has the occlusion information avaliable when it is performing classification. The context of the new occlusion algorithm with respect to existing VI is presented in figure 2.14. Boxes with solid borders represent tangible outcomes during processing. Boxes with dashed borders describe intangible processes (the logic within VI and the new algorithm). Solid arrows represent existing process flows, while the dashed arrow represents a hypothetical process flow.

Box (a) represents the image frames collected by VI for the duration of the trigger. VI continuously monitors the trigger search window for the presence of a trigger template.

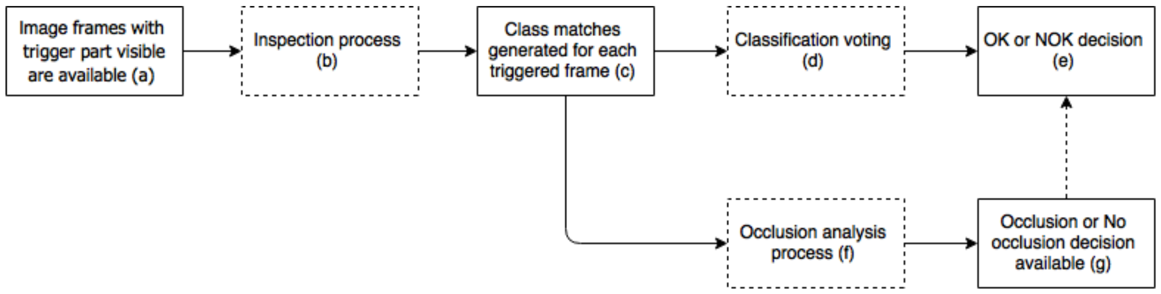
Image frames with trigger part visible are available (a)
Inspection process (b)
Class matches generated for each triggered frame (c)
Classification voting (d)
OK or NOK decision (e)
Occlusion analysis process (f)
Occlusion or No occlusion decision available (g)

Figure 2.14: Context of the new algorithm with respect to existing VI processing.

When a trigger template is detected, VI starts recording all image frames until the trigger template is no longer visible within the trigger search window. These recorded image frames are passed on to the inspection process in box (b) as a single dataset. The inspection process then checks each image frame for the presence of one of the inspection templates within the defined inspection search windows. Inspection templates corresponding to whether a part in a particular inspection window is present or not, is available to VI. VI analyzes each inspection window for the presence of all the inspection templates (or class templates) defined for that particular window, and generates class correlation values for each class associated with that inspection window. This is represented by box (c). These class correlation values are utilized by the voting process in box (d). The voting process counts the number of occurrences of each class defined within an inspection window (according to the class correlation values), and calculates which class had the maximum count. The class with the maximum count is deemed the winner by box (e), and a decision is made as to whether the inspection was "OK" (a valid class was detected within the inspection search window), or "NOK" (a valid class was not detected - this could be because the required part was not installed at all, or due to occlusion of the part during the inspection). The class correlation values represented by box (c) are also passed on to the new algorithm that performs occlusion analysis. A sample representation of the class correlation values for an inspection search window having a single inspection search window, and three different possible classes is provided in table 2.4. The occlusion analysis process represented by box (f) calculates the standard deviation values separately for each class within an inspection

Table 2.4: Sample class correlation values generated by an inspection. A sesquence of five frames have been triggered. Class 1 is the class present in the inspection search window, evidenced by its high correlation values compared to the other two classes.

Frame no.	Trigger correlation value	Inspection area no.	Class 1 correlation value	Class 2 correlation value	Class 3 correlation value
140	0.83	1	0.88	0.38	0.36
141	0.82	1	0.89	0.38	0.37
142	0.83	1	0.89	0.35	0.36
143	0.85	1	0.87	0.37	0.36
144	0.83	1	0.88	0.38	0.37
145	0.83	1	0.88	0.38	0.38

search window. The process then calculates the average standard deivation across each class within a search window, and checks if the average value is above a pre-defined threshold. If so, the process has identified an occlusion. This is represented by box (g). As part of this research work, this is as far as we have taken the process. The next step will have to be the actual integration of this new detection process into the existing VI process, and we envision the result available in box (g) to be made available to VI in time to make a correct OK or NOK decision if occlusions are detected. This is represented by the dashed line connecting the output of box (g) to the input of box (e).

2.7 Verification of the new algorithm

In order to test the validity of the new algorithm, live data collected from the BMW plant was used - video was recorded at the assembly lines at the plant, and this recorded video was then run through VI for detection and classification. VI has a special mode built-in whereby a video file can be specified as input to the inspector, rather than images from a video camera, and this mode was utilized to perform the verification.

Table 2.5: Ground-truth information for the spring coil use case (truncated to the first five cars inspected). This use case had two inspection areas, and hence, each car has two rows in the table. A '0' indicates that no occlusion occured within that particular inspection area in the video, while a '1' specifies that an occlusion was observed within that particular inspection area.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)
1	1	1
1	2	0
2	1	1
2	2	1
3	1	0
3	2	0
4	1	0
4	2	0
5	1	1
5	2	1

2.7.1 Generation of ground-truth data

As the first step of the verification process, ground-truth data was noted for each of the video files containing live data. Live data from the following five installations of VI were considered: the spring coil use case, the door-line emblem use case, the child safety lock use case, the underbody use case, and the wheel lock use case. For each video file generated from the above installations, the ground truth data for each car inspected consisted of the inspection area number and whether the inspection template within each inspection area was occluded or not. The ground-truth data was collected manually, by observing each video and noting down the occlusions as they happen. The ground truth generated for each use case, truncated to include only a few samples of the inspections, is presented in tables 2.5 to 2.9. Sample images of the inspection problem from each use case are presented in figures 2.15 to 2.20.

After recording the ground-truth data for all use cases, the total number of occlusions observed for each use case were collated into a single table (see table 2.10) to aid

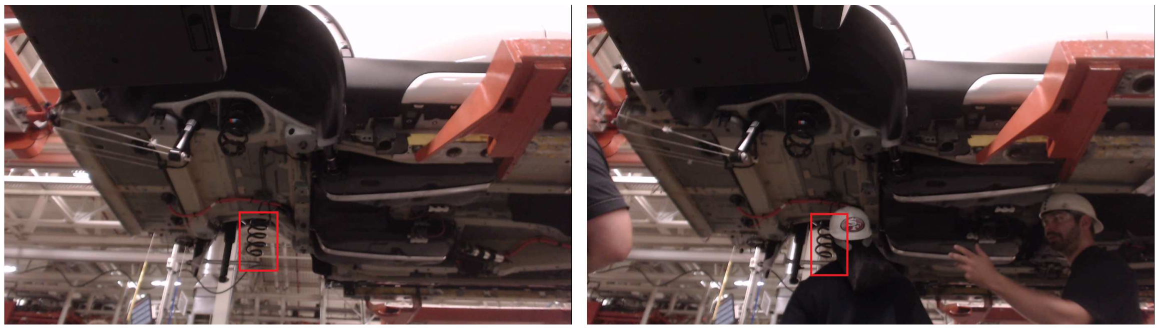

Figure 2.15: Sample images taken during the inspection process for the spring coil use case. (From left to right) case with no occlusion; case with occlusion.

Table 2.6: Ground-truth information for the door-line emblem use case (truncated to the first five cars inspected). This use case had two inspection areas, and hence, each car has two rows in the table.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)
1	1	0
1	2	0
2	1	0
2	2	0
3	1	0
3	2	0
4	1	0
4	2	0
5	1	0
5	2	0

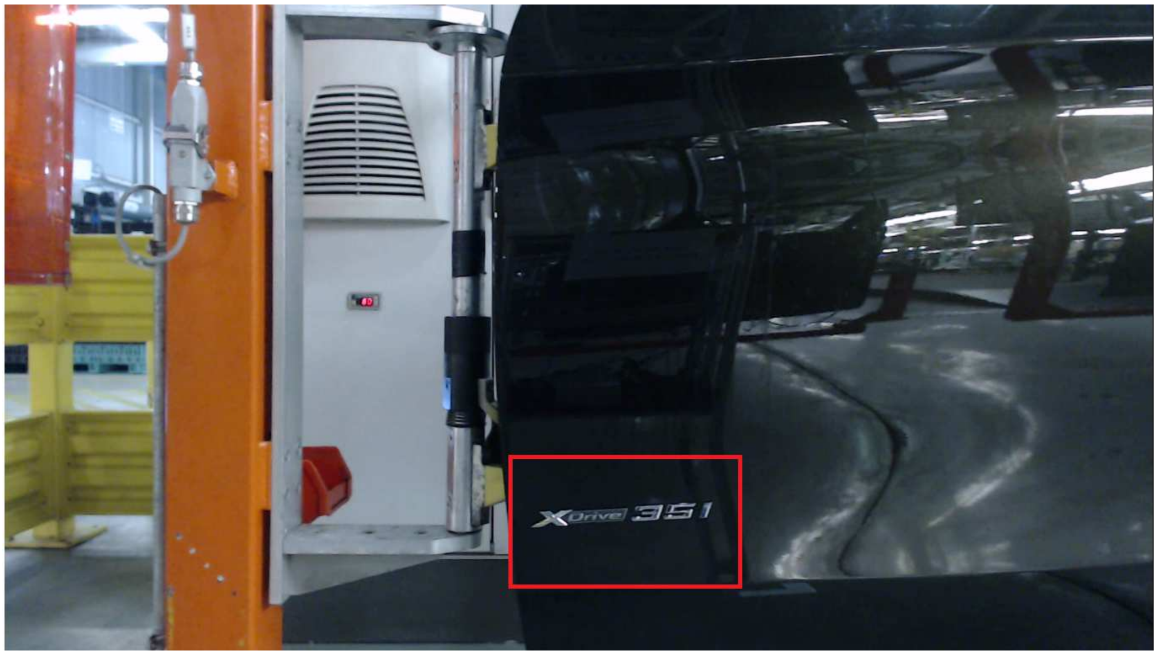

Figure 2.16: Sample image taken during the inspection process for the door-line emblem use case. This use case does not have any inspections with occlusions. The part being inspected is visible within the inspection search window.

Table 2.7: Ground-truth information for the child safety lock use case (truncated to the first five cars inspected). This use case had only a single inspection area, but each car was triggered between one and three times. This variation is due to the way this particular use case is setup in the BMW plant - the assembly (car door) appears within the trigger window momentarily, before shifting position and rising up to the worker's eye level so that additional installation of parts can be performed on the assembly. After the worker completes the installation, the assembly is then lowered back into the field of view of the VI camera, which then causes VI to trigger again. See figure 2.17 for a plot of the correlation values corresponding to multiple triggers on a single car. In three of the assemblies observed (out of a total of 72 for this use case), an additional inspection/ installation was manually performed by the workers, which resulted in the assembly being triggered a third time. Hence, for the child safety lock use case, each car has a variable number of rows in the table.

Car no.	Inspection area no.	Pass no. (no. of times triggered)	Occlusion ground-truth (0:no; 1:yes)
1	1	1	0
1	1	2	0
2	1	1	0
2	1	2	0
3	1	1	0
3	1	2	0
4	1	1	0
4	1	2	0
5	1	1	0
5	1	2	0

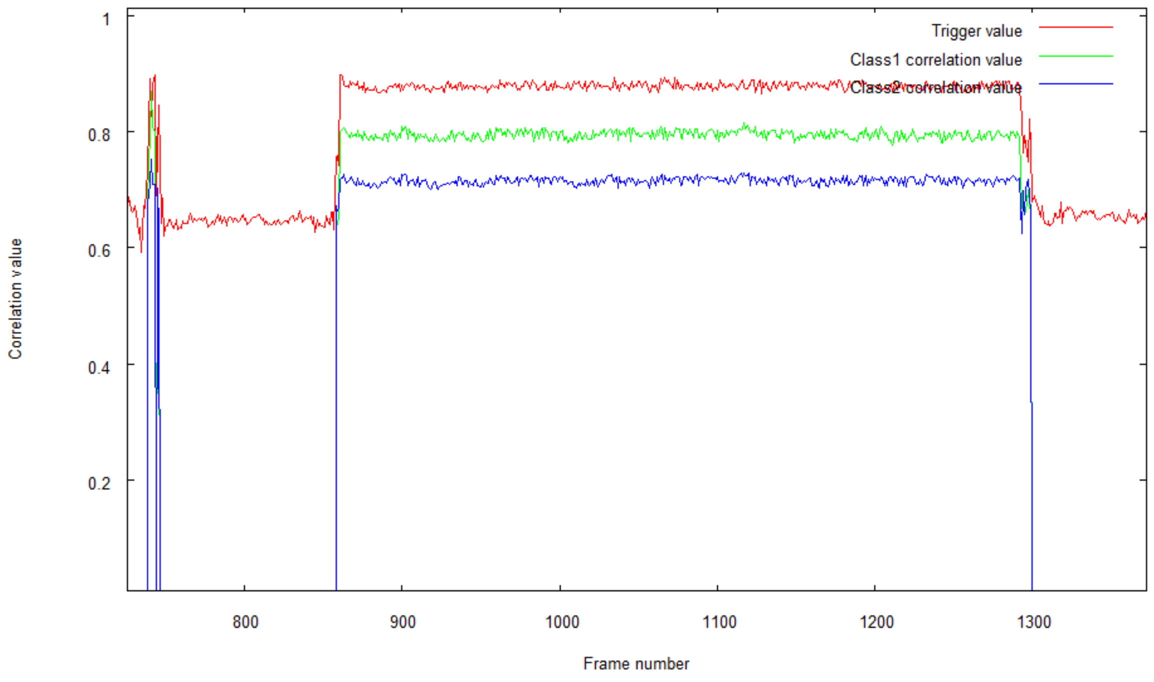

Figure 2.17: Plot of the class correlation values observed per frame for a single car in the child lock use case. The plot demonstrates a single car that triggers twice. This multiple trigger for a single car is unique to the child safety lock use case, and happens because of the way in which the use case is set up in the BMW plant.

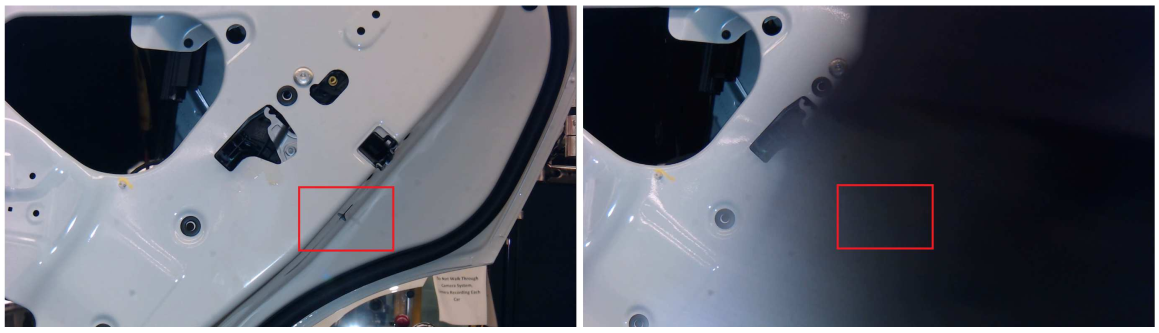

Figure 2.18: Sample images taken during the inspection process for the child safety lock use case. (From left to right) case with no occlusion; case with occlusion. The case with occlusion has also occluded the trigger, and this case (along with 7 other similar cases) will not be considered as occlusions in the ground-truth.

Table 2.8: Ground-truth information for the underbody use case (truncated to just the first car inspected). This use case was out of the ordinary, because it had four separate triggers set up (the other four use cases operate with only a single trigger). The underbody has a large number of areas to be inspected, spread out over the length of the car. The inspection areas were split up among four triggers so that each area could be inspected at relatively the same position (the center of the field of view of the camera) by VI. The first two triggers had six inspection areas each, while the last two triggers had five each, for a total of 22 inspection areas. Hence, each car has 22 rows in the table, and only the first car's ground truth information is shown here.

Car no.	Trigger no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)
1	1	1	1
1	1	2	0
1	1	3	0
1	1	4	0
1	1	5	0
1	1	6	0
1	2	1	0
1	2	2	0
1	2	3	0
1	2	4	1
1	2	5	1
1	2	6	1
1	3	1	0
1	3	2	0
1	3	3	0
1	3	4	0
1	3	5	0
1	4	1	0
1	4	2	0
1	4	3	0
1	4	4	0
1	4	5	0

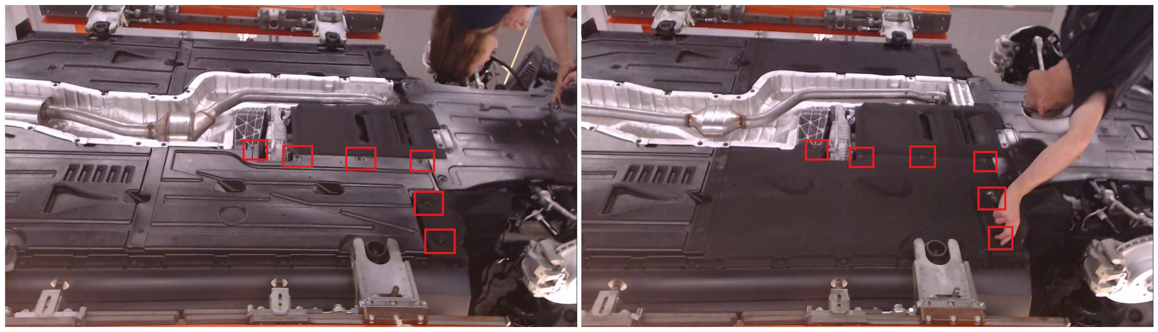

Figure 2.19: Sample images taken during the inspection process for the underbody use case. (From left to right) case with no occlusion; case with occlusion. Note: all six inspection areas for trigger #2 have been shown above.

Table 2.9: Ground-truth information for the wheel lock use case (truncated to the first five cars inspected). This use case had only a single inspection area, and hence, each car has just one row in the table.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)
1	1	0
2	1	0
3	1	0
4	1	0
5	1	0

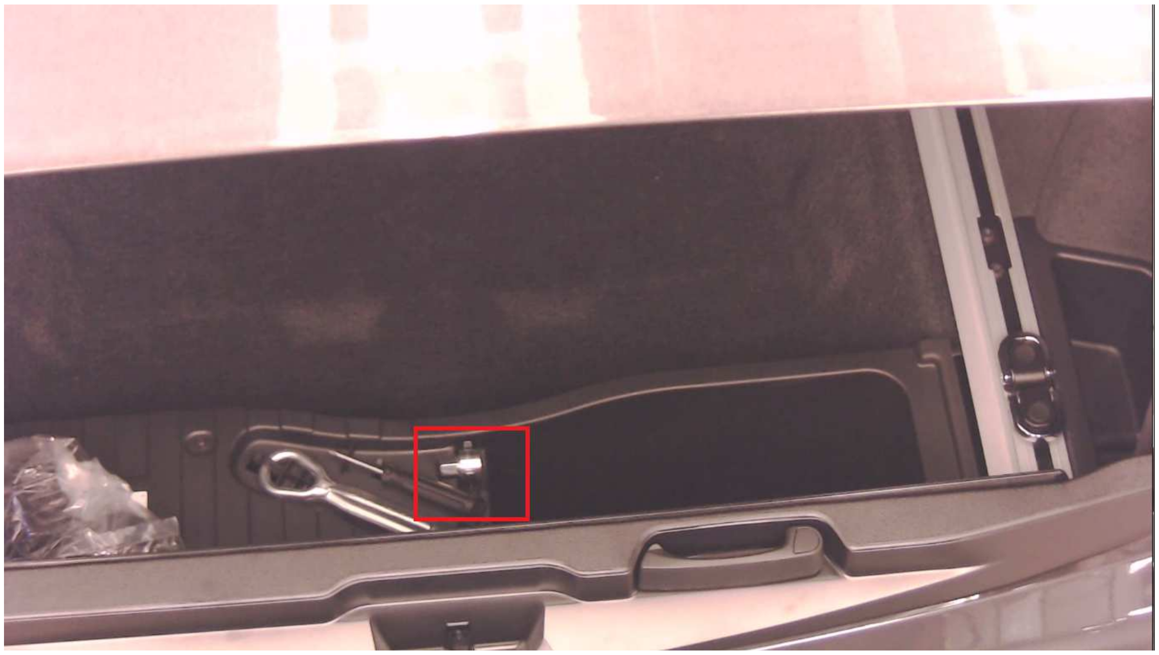

Figure 2.20: Sample image taken during the inspection process for the wheel lock use case. This use case does not have any inspections with occlusions. The part being inspected is visible in the inspection search window.

Table 2.10: The net occlusions ground-truth data.

Use case	No. of inspection areas per car	Total inspections performed	No. of occlusions (camera was blocked)	No. of occlusions (light was blocked)	Total no. of occlusions (ground-truth)
Testbed simulation	1	20	16	0	16
Spring coil	2	150	25	1	26
Door-line emblem	2	154	0	0	0
Under- body	22	264	16	44	60
Child safety lock	1	125	0	0	0
Wheel lock	1	19	0	0	0

Table 2.11: Cases where the trigger was occluded in VI.

Use case	Trigger occluded (inspection performed)	Trigger occluded (never triggered)
Underbody	0	22
Child safety lock	8	0
Wheel lock	0	10

the verification of the occlusion detection algorithm. While recording the ground-truth information, three use cases were found to have multiple inspections where the trigger was occluded and the car was either not inspected at all, or the occlusion was not detected. This information is presented in table 2.11. At the plant, VI would be connected to BMW's vehicle data base, and if a vehicle is supposed to be inspected, but was not triggered, VI can easliy flag an error. Since the vehicle database was not available for this research work, this functionality was switched off. If the trigger is occluded during inspection, VI does not generate correlation values, and there is no data to analyze. VI isn't designed to handle these cases, and the cases are not considered while calculating the detection accuracy or the overall accuracy of the new algorithm.

2.7.2 Determining the threshold value of standard deviations for detecting occlusions

The second step of the verification process was to determine the actual threshold for standard deviation values. The algorithm detects occlusions by checking whether the standard deviation of the correlation values exceeds this threshold. If the threshold is breached, an occlusion is said to have occurred.

Live data from the BMW plant (the same data that was used to generate the ground-truth information for the six use cases in table 2.10) was introduced to VI, and for each use case, the class correlation values generated were recorded. This recorded information was then analyzed, and the standard deviation values were calculated. These values were then matched up to the right ground-truth inspection for every use case. Hence by the end of this process, every single ground-truth data had a standard deviation value (of the corresponding class correlation values) attached to itself. Next, histograms of the standard deviation values were generated separately for ground-truth cases with and without occlusions. For each use case, the histograms were then superimposed, to determine whether a clear separation could be observed between cases with and without occlusions. Ideally, cases without occlusions would have standard deviation values exclusively at the lower end of the scale, whereas occlusions would register much higher standard deviations, with no overlapping values between the two types of cases. A threshold value of standard deviation could thus be picked, in order to complete the occlusion detection process. If no clear separation was observed, the idea was to then pick a threshold value low enough that would maximize the potential for detecting occlusions (with the disdvantage of possibly a larger number of false positive results).

2.7.3 Inspection of live data with the new algorithm and standard deviation threshold values

As the final step of the verification process, the standard deviation threshold values for each use case obtained from the previous step were utilized within the new algorithm in order to determine whether or not occlusions had occurred. The results of this step were cross-verified with the ground-truth information to determine the accuracy of the new algorithm.

Chapter 3

Results

3.1 Analysis of correlation values generated for the different types of occlusions simulated on the testbed

Simulations were run to generate the correlation values corresponding to each different type of occlusion. After these values were generated by the modified VI, the standard deviations of each inspection were calculated. This is presented in table 3.1. As expected, the standard deviation is very low when there is no occlusion, or when there is a partial occlusion. For brief and intermittent occlusions, however, we can see a measurable increase in the amount of deviation.

Table 3.1: Observed standard deviation values for simulated occlusions.

Type of occlusion	Standard deviation value (σ)
None	0.01
Brief	0.066
Partial	0.008
Intermittent	0.062

3.2 Selection of the standard deviation threshold values for each use case

3.2.1 Selection of the threshold value for the testbed simulation

Figure 3.1 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the testbed simulation. Two different histograms are shown here - one set for ground-truth cases with no occlusions (in red), and another set for ground-truth cases with occlusions (in green). The standard deviation value observed is presented along the X-axis, and the frequency of occurrence of the standard deviation values is presented along the Y-axis. According to the ground-truth, this use case had four inspections with no occlusion, and 16 inspections with occlusion. As can be observed, cases with no occlusions have low standard deviation values, signified by the red bars clustering towards the left side of the graph. On the other hand, cases with occlusion display significantly higher standard deviation values on average. A point to note here is the overlap between cases with and without occlusion at the standard deviation values 0.008 and 0.009. The three cases with occlusion having these low values of standard deviation were instances of partial occlusion - VI generates consistent correlation values for these type of occlusions, and the algorithm is not designed to detect these cases. Hence, even though the ground-truth information for these cases say that an occlusion has occured, the standard deviation values generated say otherwise. The solitary ground-truth occlusion case at standard deviation 0.006 is also a partial occlusion, and is not detected by the algorithm for the same reasons detailed above. The plot of correlation values generated per frame for these four outlier cases are presented in figure 3.2.

The remaining cases in this use case demonstrate very good separation between occlusions and no occlusions. This separation is as we expected, and this makes choosing a threshold value simpler. Ultimately, a standard deviation threshold value of 0.040 was chosen for this use case, as it accurately distinguishes between all true positives (barring the cases with partial occlusion) and true negatives.

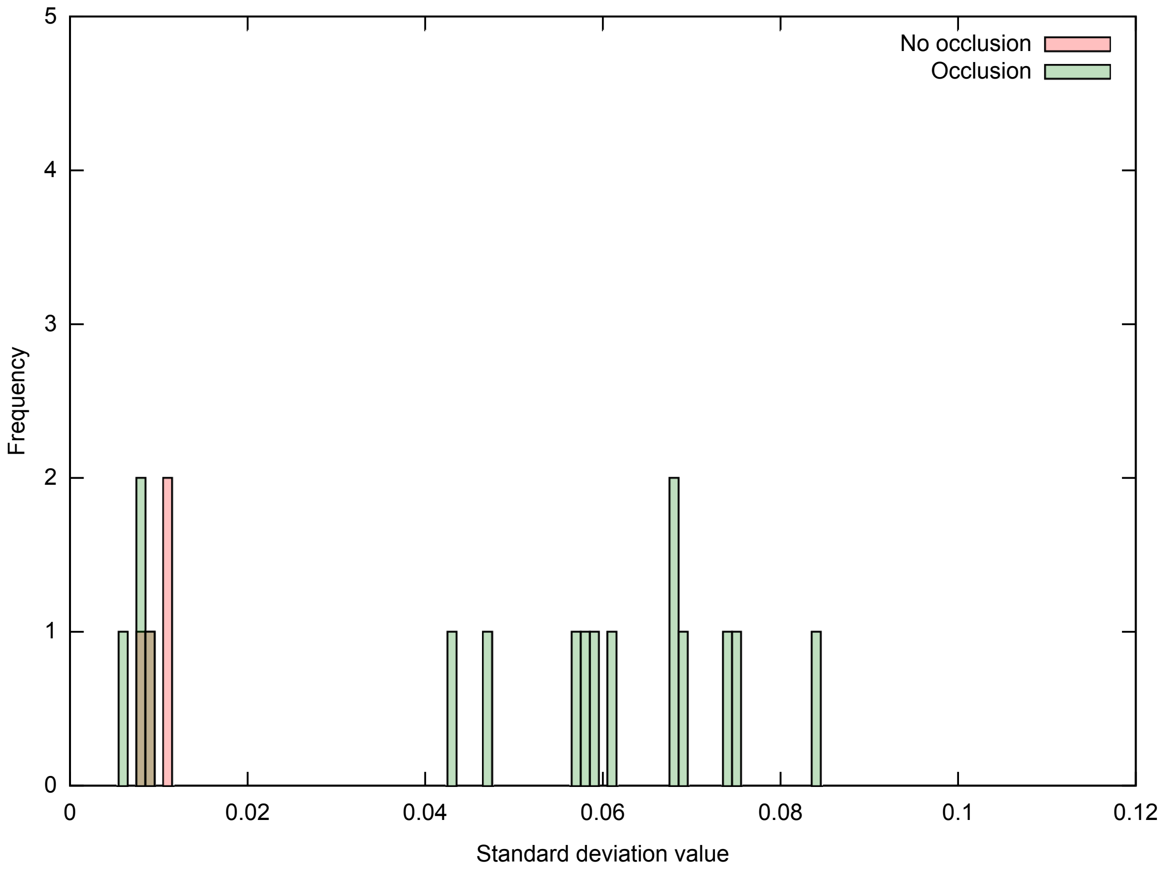

Figure 3.1: Histogram of observed standard deviation values corresponding to the ground-truth data for the testbed simulation (classified according to whether an occlusion was observed or not).

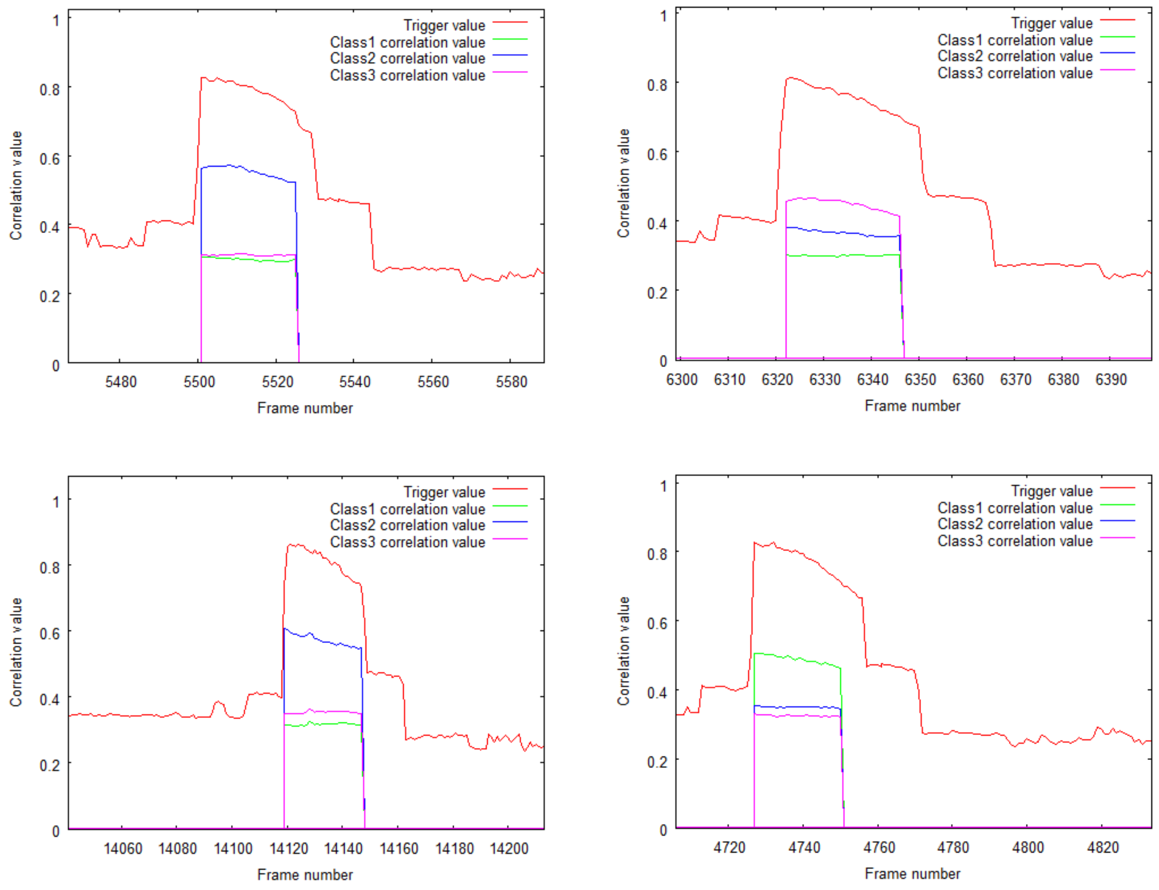

Figure 3.2: Plots of class correlation values observed per frame for each of the four partial occlusions part of the testbed simulation. In each case, the class actually being inspected has the highest correlation values, while the other two classes have lower (but non-zero) correlation values. The consistent nature of the correlation values means that they have low standard deviation values as well, and such cases are not detected by the new algorithm.

3.2.2 Selection of the threshold value for the spring coil use case

Figure 3.3 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the spring coil use case. As for the previous use case, ground-truth cases with no occlusions are presented in red, and ground-truth cases with occlusions are in green. According to the ground-truth, this use case had 124 inspections with no occlusion, and 26 inspections with occlusion. Cases with no occlusions are clustered mainly at the left side of the graph. However, cases with occlusion are more widely dispersed in this use case compared to the previous one. There are quite a few points of interest in this particular histogram: the overlap between four cases with occlusion and other cases with no occlusion at standard deviation values 0.008, 0.009, 0.018, and 0.020.

The three cases with occlusion having values of standard deviation 0.008, 0.010, and 0.020 were instances of partial occlusion - the inspection areas were completely occluded by a worker for the entire duration of the inspection. Hence, the corresponding class correlation values for these inspections were very stable throughout the inspection. These cases will not be detected by the new algorithm. The plot of correlation values generated per frame for these three cases are presented in figure 3.4.

The fourth ground-truth occlusion case at standard deviation 0.018 was a brief occlusion, but VI generated consistent correlation values throughout the duration of inspection. This is a false negative, and is not detected by the algorithm because of the low standard deviation values. The plot of correlation values generated per frame is in figure 3.5.

Unlike the histogram of the testbed simulation, there is no clear separation between the cases with and without occlusions in the spring coil use case. Occlusion cases with the standard deviation values closest to the highest value recorded for a non-occlusion case were again verified manually by observing the video file, as well as the correlation value trace. All such cases were found to be true occlusions, and the analysis of the trace for four of the occlusion cases with the least standard deviation values are presented in figure 3.6.

For the spring coil use case, the separation between the cases with and without occlusion is very little, and this makes choosing a threshold value a little more difficult.

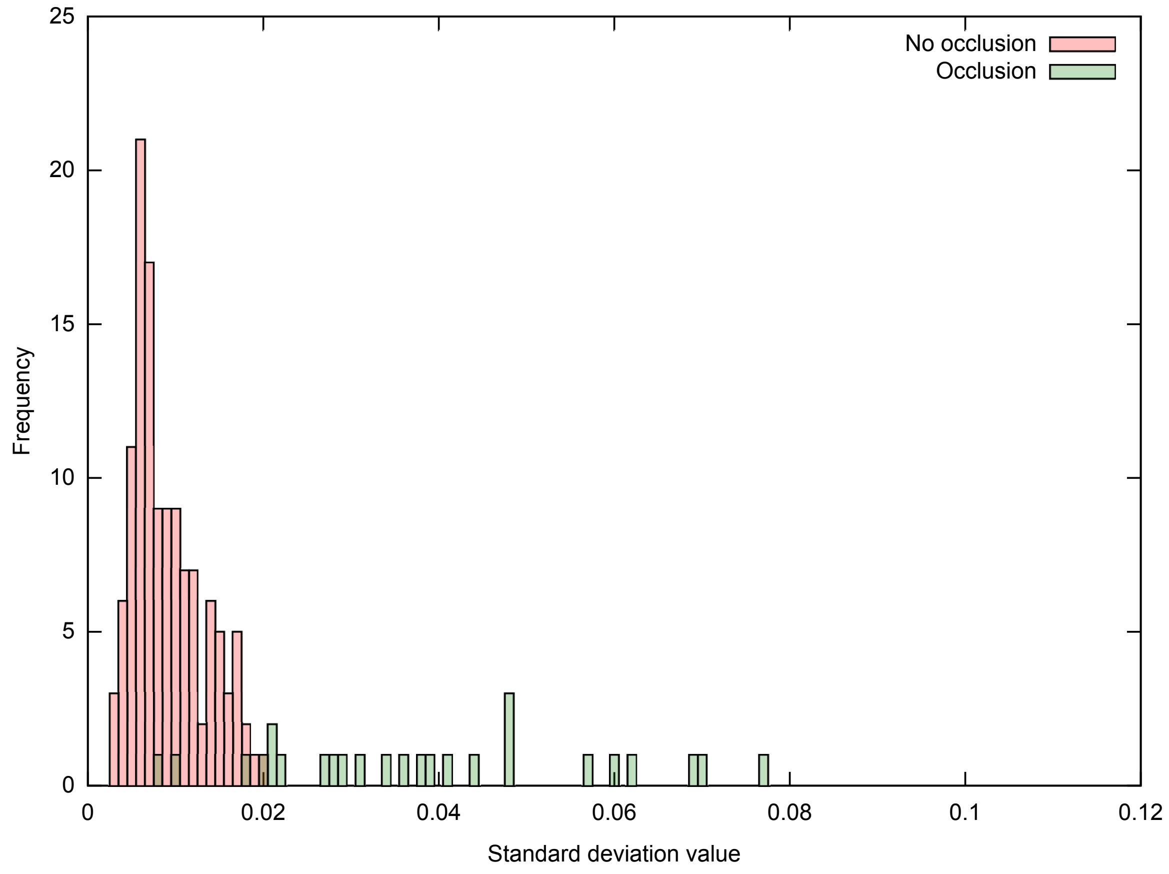

Figure 3.3: Histogram of observed standard deviation values corresponding to the ground-truth data for the spring coil use case (classified according to whether an occlusion was observed or not).

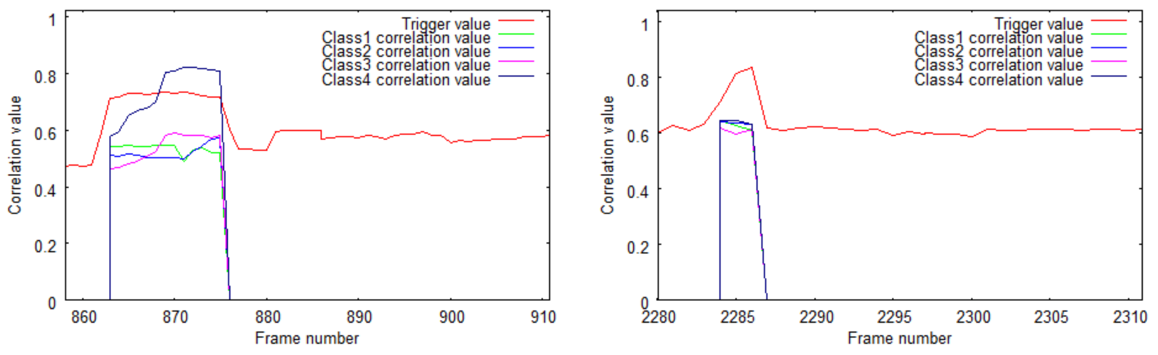

Figure 3.4: Plots of the class correlation values observed per frame for the three instances of partial occlusion in the spring coil use case. The spring coil use case has two inspection areas, with two classes per inspection area, for a total of four classes as depicted in the plots above. For the plot on the left, the false negative occured in only one inspection area - denoted by the light green and light blue lines. The occlusion in the second inspection had a marked effect on the standard deviation for the respective classes in that area, as evidenced by the large fluctuation in the pink and dark blue lines. For the plot on the right, the partial occlusion affected both inspection areas equally, and hence the relatively similar traces for all four classes.

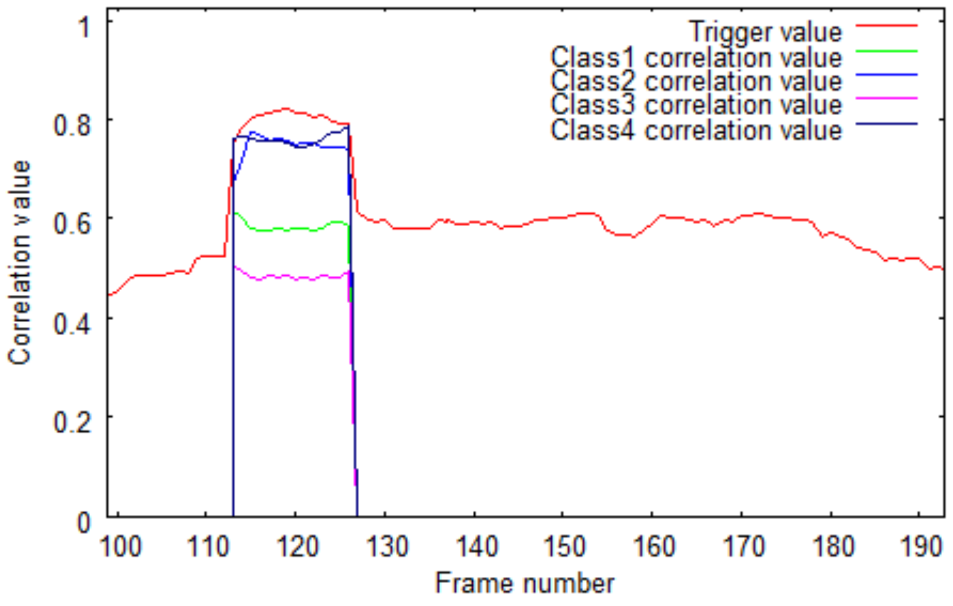

Figure 3.5: Plots of the class correlation values observed per frame for the brief occlusion not detected in the spring coil use case. The occlusion happened on the first inspection area, right at the begining of the inspection process. This is evidenced by the upward spike in the trace corresponding to Class2 (the light blue line). However, the average standard deviation of the correlation values within this inspection area was still quite low.

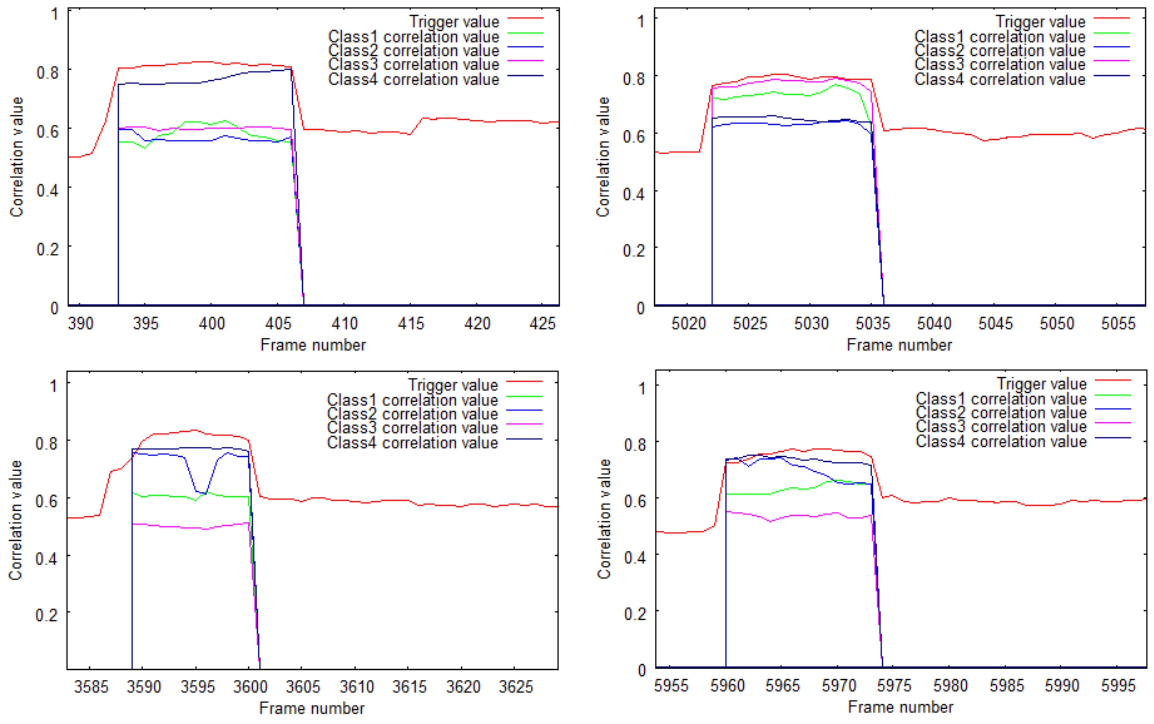

Figure 3.6: Plots of the class correlation values observed per frame for the four occlusions with lowest standard deviation values. (clockwise from top left) standard deviation value 0.021 - occlusion of the spring in the first inspection area (the fluctuation of the green line); standard deviation value 0.021 - inspection of a missing spring in the first inspection area, where the worker occludes the missing part right at the end of inspection (the momentary dip in the green line at the end); standard deviation value 0.027 - occlusion of the spring in the first inspection area (the dip in the light blue line during the second half of the inspection); standard deviation value 0.028 - brief occlusion of the spring in the first inspection area (the trough visible on the light blue line).

Ultimately, a standard deviation threshold value of 0.020 was chosen for this use case, as it adequately distinguishes between true positives (barring the cases with partial occlusion) and true negatives. It must be noted that this threshold value is lesser than that chosen for the testbed simulation use case. This is because the standard deviation values observed for true occlusions in the spring coil use case are lower than those observed for the testbed simulation. A possible explanation for this may lie in the different way the two use cases are set up. The testbed necessitates placing the camera very close to the subject being inspected. Consequently, even a small occlusion fills up a large portion of the field of view of the camera. Thus, any movement captured will have a large impact on the correlation values that VI generates. This is not so in the case of the spring coil use case at the BMW plant. The camera is much farther away from the car, with a much larger field of view. Therefore, the use case is slightly less sensitive to changes in the image during inspection, which translates to less fluctuations in the correlation values, and consequently, less standard deviation values. Moreover, the fluidity of movement of the assembly line also has to be taken into consideration. The testbed assembly is much less smoother than the spring coil assembly line, which may contribute to larger fluctuations in correlation values.

3.2.3 Selection of the threshold value for the door-line emblem use case

Figure 3.7 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the door-line emblem use case. The ground-truth cases without occlusions are presented in red, and there are no cases with occlusions for this use case. According to the ground-truth, this use case had 154 inspections in total, all with no occlusion. Most of the inspections with no occlusions are clustered at the left side of the graph. However, a particular inspection shows a relatively high standard deviation value of 0.047. The analysis of the trace for this inspection is presented in figure 3.8.

For the door-line emblem use case, there was no need to consider separation between the cases with and without occlusion. Ultimately, a standard deviation threshold value of 0.050 was chosen for this use case, to accommodate all true negatives. It must be noted that

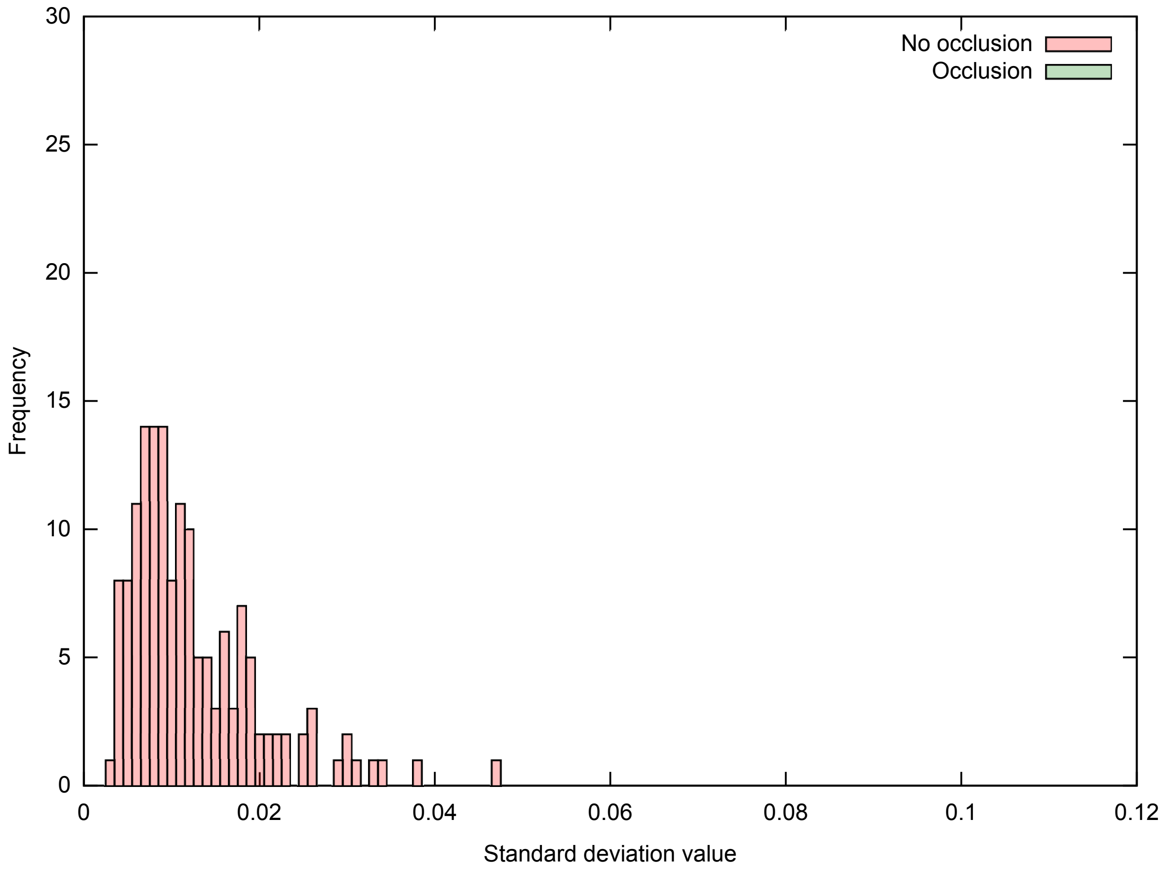

Figure 3.7: Histogram of observed standard deviation values corresponding to the ground-truth data for the door-line emblem use case (classified according to whether an occlusion was observed or not). This use case does not have any occlusions.

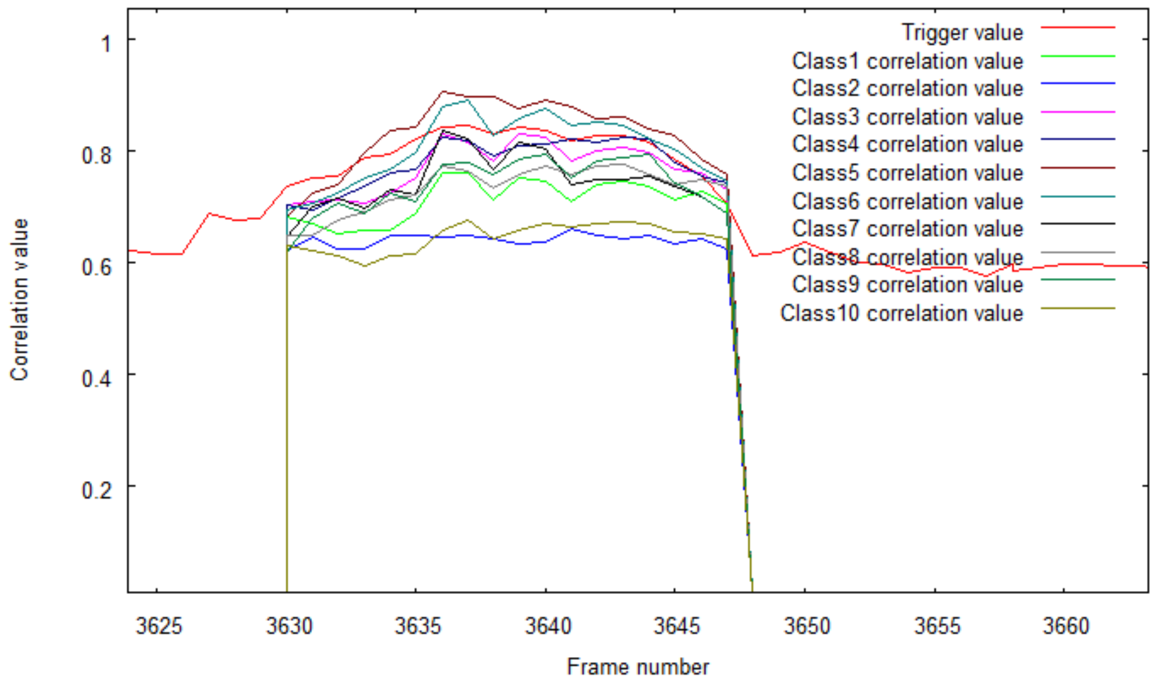

Figure 3.8: Plot of the class correlation values observed per frame for the case with no occlusion with high standard deviation value. The doorline-emblem use case has two inspection areas, with three classes in the first, and seven classes in the second area. This particular case demonstrates relatively high standard deviation values, even though there is no occlusion. After reviewing this case in the video file, it was determined that the high standard deviation value was caused by excessive shaking of the door-line assembly during inspection. The lack of a stable image contributed to high fluctuations in the class correlation values.

this threshold value is greater than that chosen for the spring coil use case. This is because the standard deviation values observed for cases with no occlusions in the door-line emblem use case are higher than those observed for the spring coil use case. Again, this difference is likely due to the different manner in which the two use cases are set up. The door-line emblem use case positions the camera closer to the subject than that done for the spring coil use case. Consequently, even small movements in the video will have a large impact on the correlation values that VI generates. Moreover, the door-line emblem assembly is a little less smoother than the spring coil assembly line, with frequent shaking of the assembly during inspection that may in turn contribute to larger fluctuations in correlation values.

3.2.4 Selection of the threshold value for the child safety lock use case

Figure 3.9 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the child safety lock use case. Ground-truth cases with no occlusions are presented in red, and there are no cases with occlusion. According to the ground-truth, this use case had 125 inspections with no occlusion. The histogram for this use case is not as expected - cases with no occlusions are not limited to the left side of the graph, quite a few cases exhibit high standard deviation values. There are a lot of points of interest here: the large number of inspections with no occlusion occurring at relatively high standard deviation values greater than 0.040.

Eight cases with no occlusion demonstrated higher than expected values of standard deviation that were greater than 0.040. Each of these cases were manually reviewed, and it was verified that there were indeed no occlusions. The reason for the high standard deviations for each of these inspections was because of excessive shaking of the assembly during inspection that caused large fluctuation in the class correlation values. Similar to the door-line emblem use case, the child safety lock use case positions the camera very closer to the subject being inspected. Consequently, tiny movements in the video translate to large changes in the correlation values that VI generates. Moreover, the child safety lock assembly is a little less smoother than the spring coil assembly line, with frequent shaking

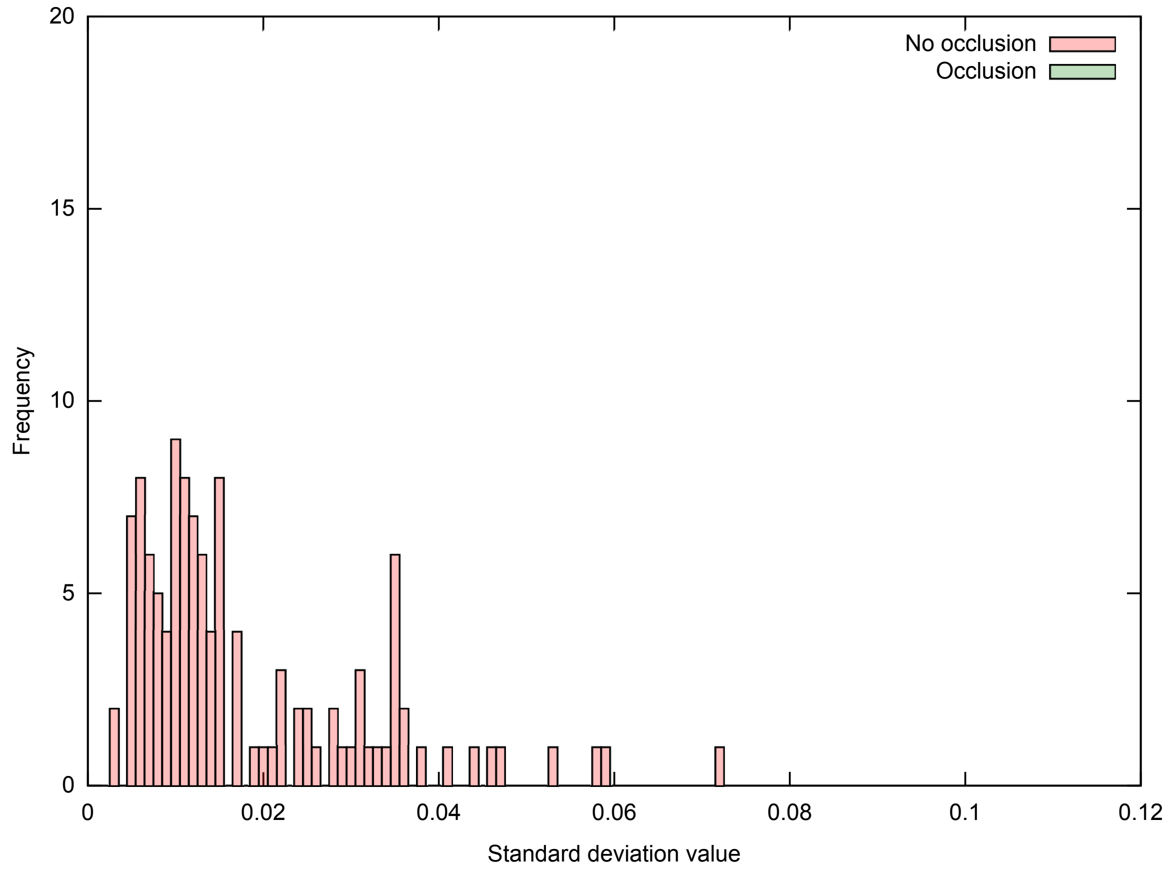

Figure 3.9: Histogram of observed standard deviation values corresponding to the ground-truth data for the child safety lock use case (classified according to whether an occlusion was observed or not).

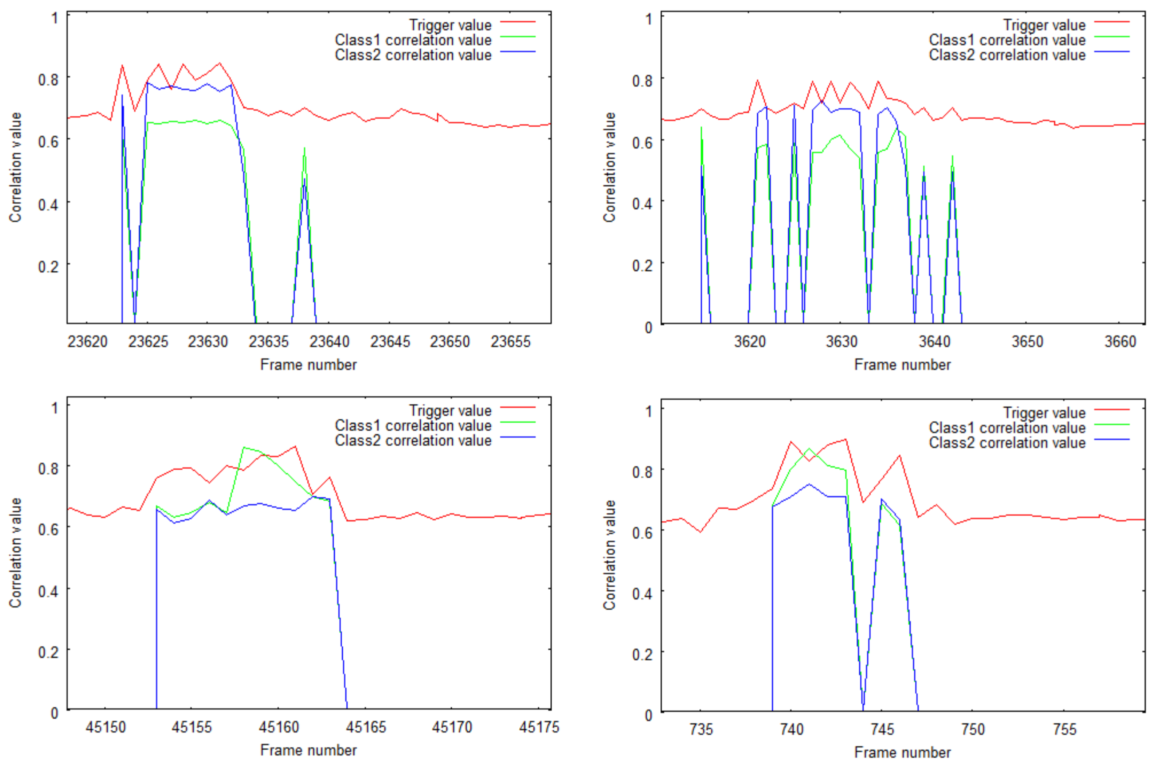

Figure 3.10: Plots of the class correlation values observed per frame for the first four cases with no occlusion having high standard deviation values in the child safety lock use case. It is important to note that although the trigger value (the red line) falls below the trigger threshold for three of the four cases above, the corresponding zero-valued class correlations (the light blue and light green lines) are not considered while calculating the standard deviation. The class correlation values demonstrate sufficiently high fluctuations by themselves to result in high standard deviation values.

of the assembly during inspection that may in turn also contribute to larger fluctuations in correlation values. The plot of correlation values generated per frame for these eight inspections are presented in figure 3.10 and figure 3.11.

There were eight special cases (not part of the 125 inspections considered as part of this use case), where the trigger was occluded, in addition to the inspection area. The setup of the child safety lock use case in the BMW plant places a trigger window very close to the inspection window. An occlusion in the inspection window will almost always occlude the trigger window as well. However, if the trigger is occluded VI does not generate class correlation values at all, and hence, there is no data to analyze for the few frames where

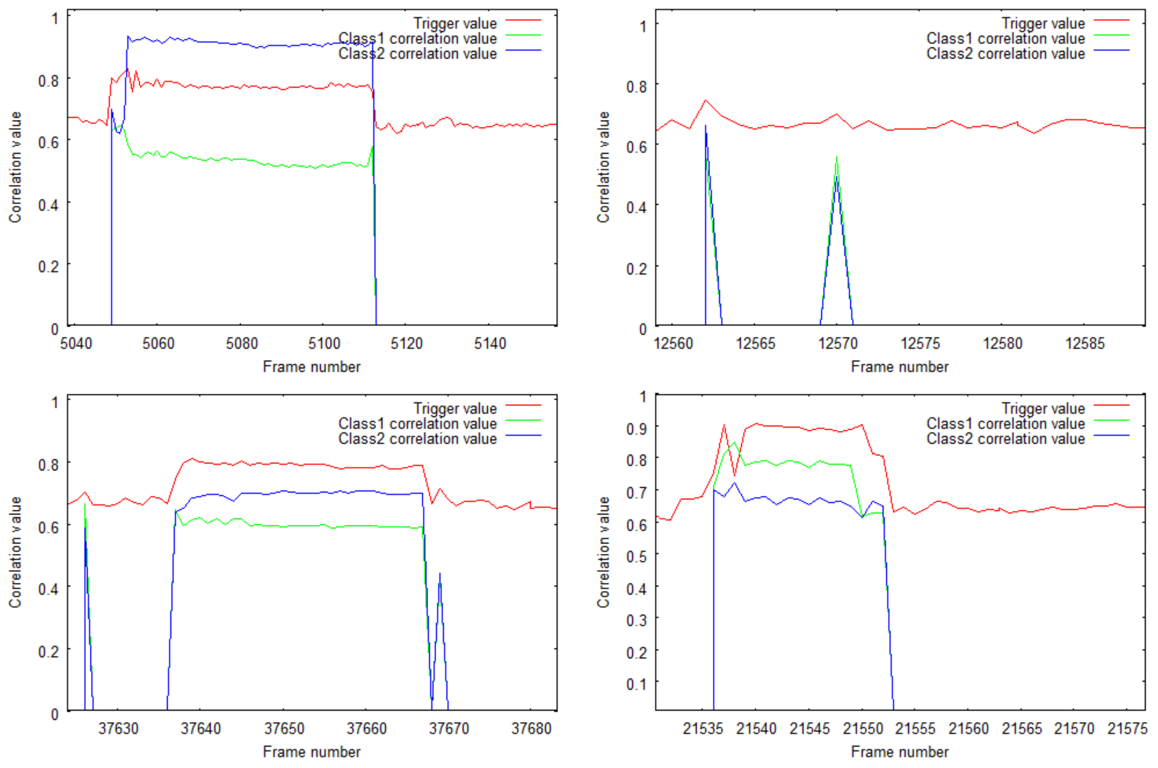

Figure 3.11: Plots of the class correlation values observed per frame for the second four cases with no occlusion having high standard deviation values in the child safety lock use case. Again, it is important to note that zero-valued class correlations above are not considered while calculating the standard deviation. The class correlation values demonstrate sufficiently high fluctuations by themselves to result in high standard deviation values.

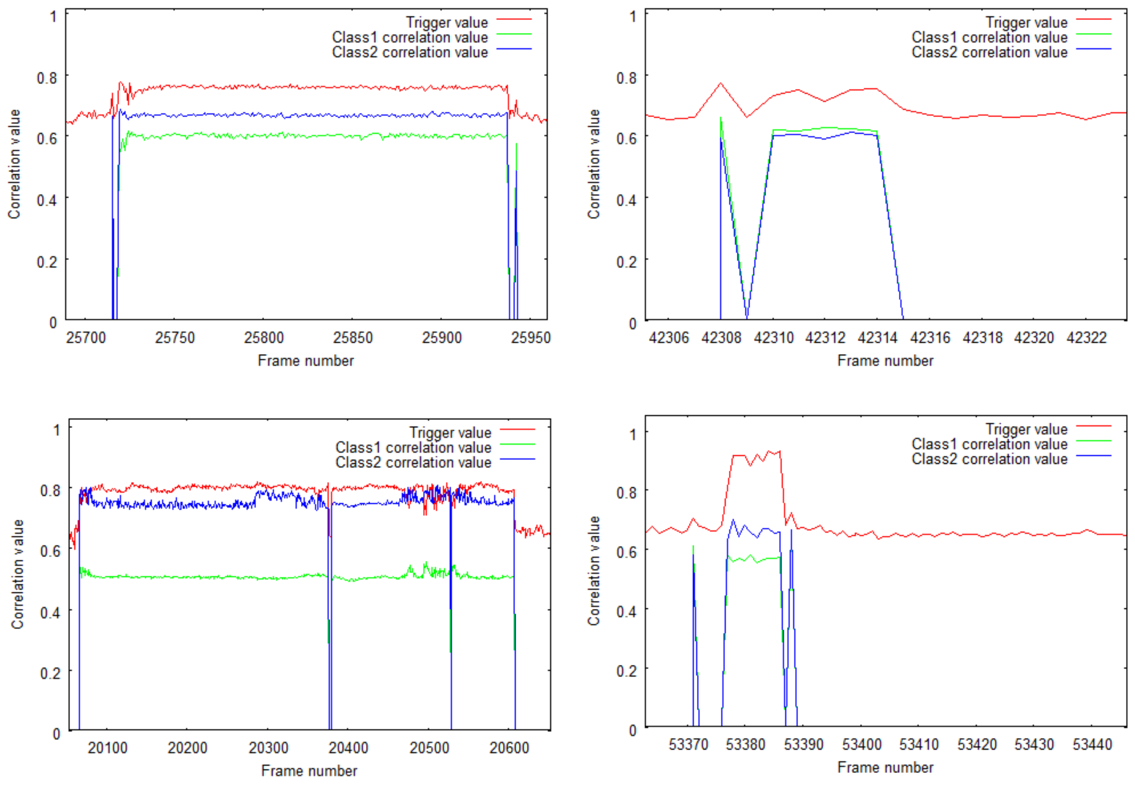

Figure 3.12: Plots of the class correlation values observed per frame for the first four instances where the trigger was occluded in the child safety lock use case. The child safety lock use case has a single inspection area with two classes. All the cases depicted above had occlusions that occluded the trigger, in addition to the inspection area. These are characterised by the trigger value (the red line) falling below the threshold of 0.70, and the corresponding class correlation values (the light blue and light green lines) dropping down to zero. These cases are presented for information only - they are not included for analysis by the new algorithm.

the occlusion causes the trigger value to fall below the trigger threshold. Hence, zero-valued class correlations for the frames corresponding to when the trigger was occluded are not considered as part of the occlusion analysis. The current setup of VI is not designed to accommodate inspections where the trigger is occluded. Hence, such cases are not considered for analysis by the new algorithm. The plot of correlation values generated per frame for these eight inspections are presented in figure 3.12 and figure 3.13.

For the child safety lock use case, there is no measure of separation between the cases with and without occlusion. Cases with no occlusion were registering very high standard

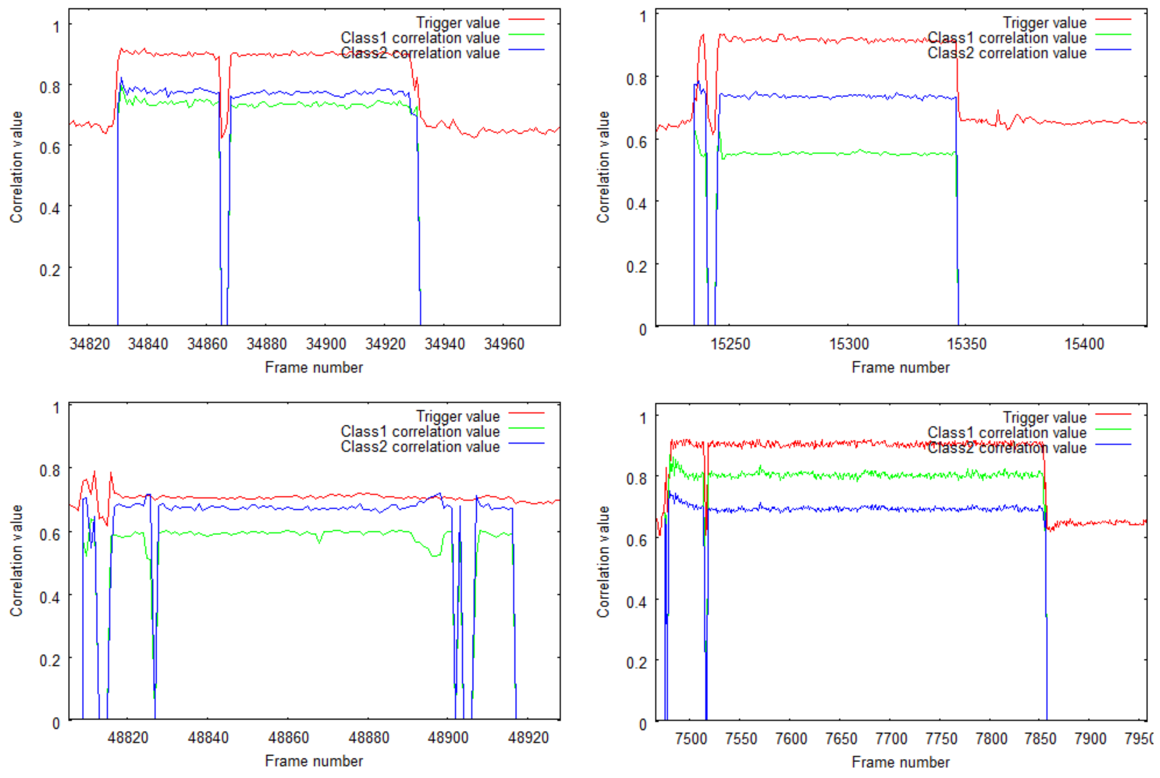

Figure 3.13: Plots of the class correlation values observed per frame for the second four instances where the trigger was occluded in the child safety lock use case. Similar to the previous four cases of occlusion, all the cases depicted above had occlusions that occluded the trigger, in addition to the inspection area. Again, these cases are presented for information only - they are not included for analysis by the new algorithm.

deviation values, because of way the use case is set up at the plant. Ultimately, a standard deviation threshold value of 0.075 was chosen for this use case, in order to capture all true negatives in this use case. It must be noted that this threshold value is much higher than that of any other use case. This is because the standard deviation values observed for cases with no occlusions in the child safety lock use case are very high. The reason for this high value is similar to the reason why the door-line emblem use case also employs a high threshold value - the camera is placed very close to the subject, which contributes to larger fluctuations in correlation values, and the assembly movement is not as smooth as that for the other use cases.

3.2.5 Selection of the threshold value for the underbody use case

Figure 3.14 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the underbody use case. Ground-truth cases with no occlusions are presented in red, and ground-truth cases with occlusions are in green. The underbody use case is unique amongst the use cases within VI, as it is the only one with more than one trigger. This use case has four triggers, with multiple inspection areas divided amonst them. The first two triggers have six inspection areas each, while the last two triggers have five each, for a total of 22 inspection areas. According to the ground-truth, this use case had 204 inspections with no occlusion, and 60 inspections with occlusion. The histogram for this use case is as expected - cases with no occlusions are clustered on the left side of the graph, and cases with occlusions are clustered on the right.

For the underbody use case, there is hard separation between the cases with and without occlusion. The standard deviation of occlusions is relatively low in this use case, because similar to the spring coil use case, the camera is far away from the subjects being inspected, and the movement of the assemmbly is fluid. Ultimately, a standard deviation threshold value of 0.020 was chosen for this use case, in order to capture all true positives and true negatives.

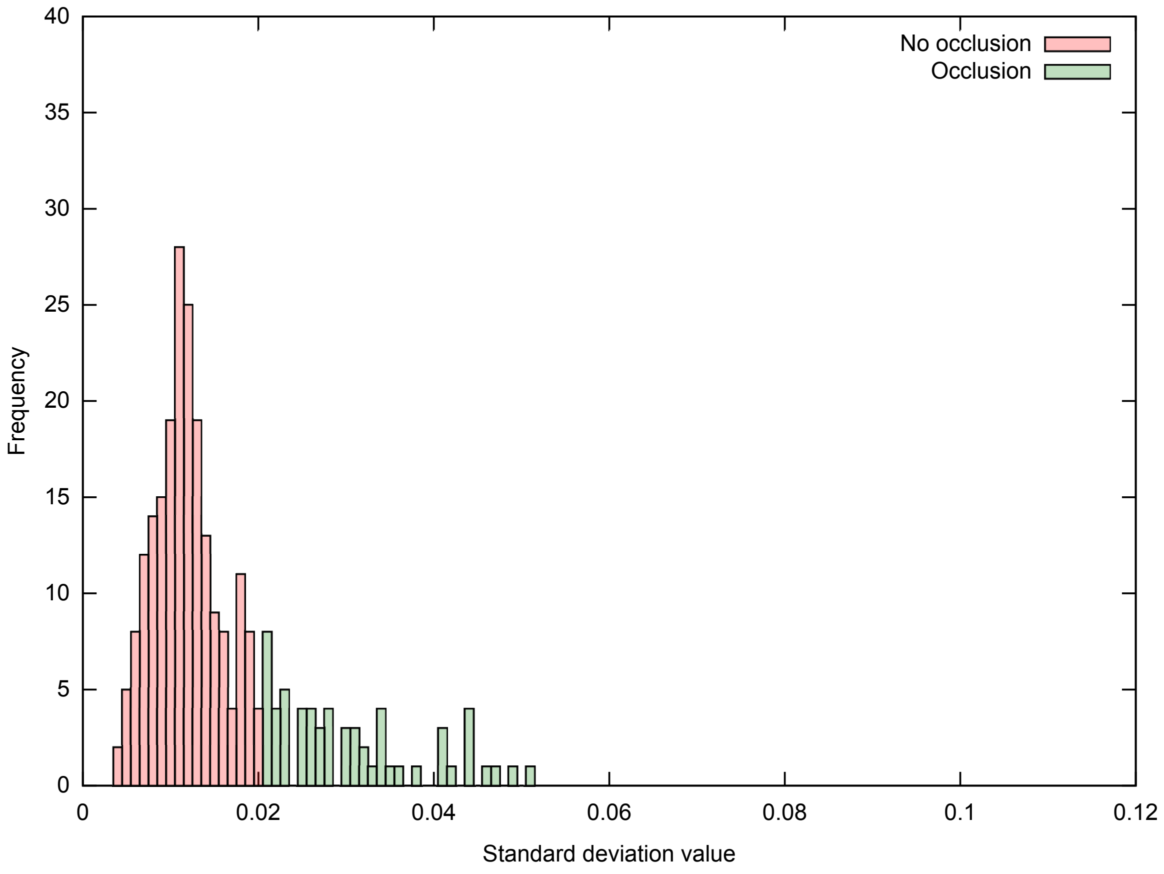

Figure 3.14: Histogram of observed standard deviation values corresponding to the ground-truth data for the underbody use case (classified according to whether an occlusion was observed or not).

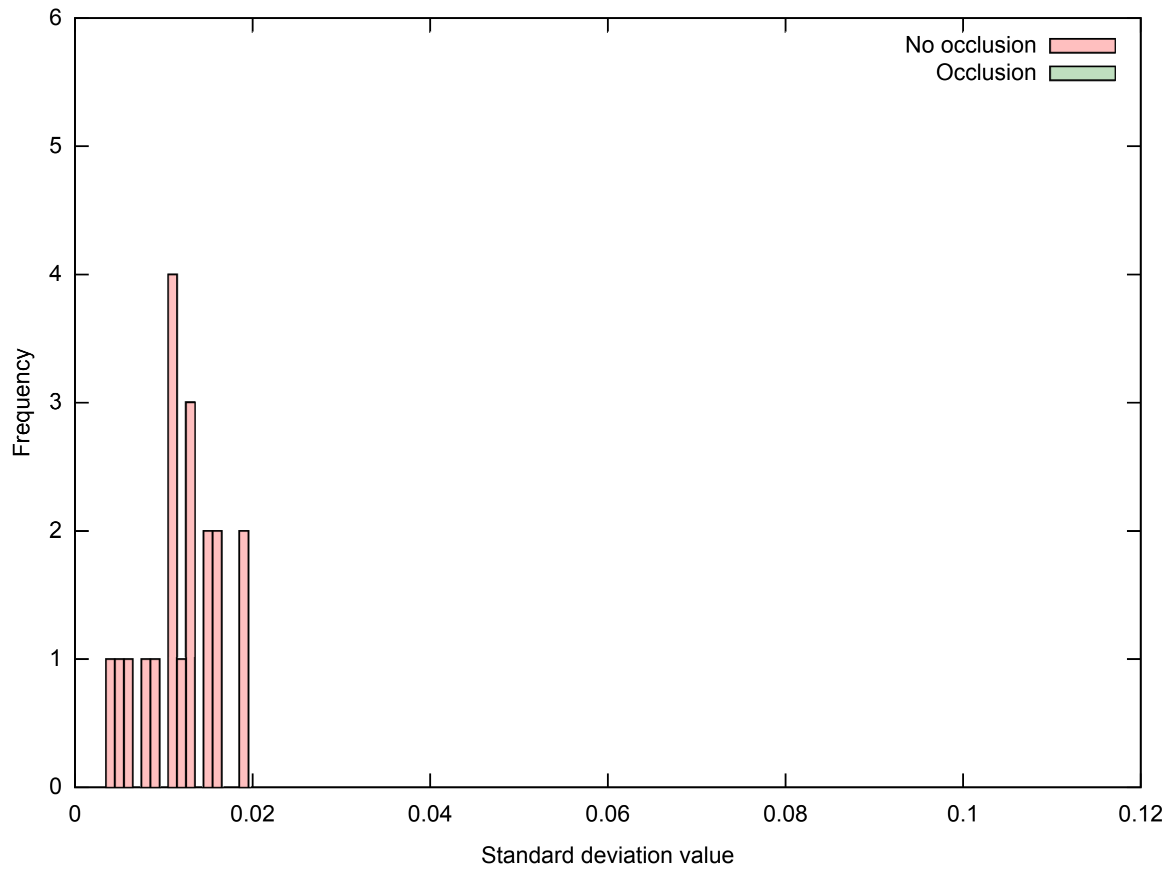

Figure 3.15: Histogram of observed standard deviation values corresponding to the ground-truth data for the wheel lock use case (classified according to whether an occlusion was observed or not). This use case does not have any occlusions.

3.2.6 Selection of the threshold value for the wheel lock use case

Figure 3.15 shows the histogram of standard deviation values observed corresponding to the ground-truth data for the wheel lock use case. The ground-truth cases without occlusions are presented in red, and there are no cases with occlusions for this use case. According to the ground-truth, this use case had 19 inspections in total, all with no occlusion. All of the inspections with no occlusions are clustered at the left side of the graph.

The inspections in this use case demonstrate good clustering. As there were no cases with occlusions, a measure of separation cannot be determined. However, the cases with no occlusion demonstrate standard deviation values as expected, and this makes choosing a threshold value simpler. Ultimately, a standard deviation threshold value of 0.040, the same as the threshold chosen for the testbed simulation, was selected for this use case as it accurately detects all true negatives. This threshold could be taken down to as low as 0.020, with no change in the final results.

3.2.7 Summary of standard deviation threshold values selected for each use case

Table 3.2 collates the standard deviation threshold values selected for all the use cases over the previous few sections. It is immediately apparent that the threshold value varies between the use cases. There are a few reasons why this is so. Firstly, lighting conditions at a particular installation can determine the quality of detection/ classification performed by VI. Better lighting will generally result in more consistent correlation values. Shadows appearing across trigger or inspection windows in particular, cause noticeable changes in the class correlation values, and can lead to false positives if the threshold value is kept too low. Secondly, the level of zoom of the camera or the distance between the camera and the inspection area matters. If the use case has a narrow field of view, small movements or motion in the video feed will have an amplified effect on the correlation values generated. Conversely, in a use case with a large field of view where the inspection area is further away fom the camera, small movements are less likely to be picked up, and less likely to affect the correlation values during inspection. Use cases with a small field of view in general tend to need higher threshold values in order to not generate false positive results. Lastly, the amount of movement or jitter in the assembly during inspection plays a large role in the amount of fluctuation in the correlation values. Use cases with stable assemblies experience correlation values that are more consistent, and can thus handle a lower threshold value to detect occlusions.

Essentially, the way a particular use case is set up at the plant will determine the class correlation values that VI generates as part of inspections for that use case. This means that similar types of inspections (ones with occlusions, and ones without) at dif-

Table 3.2: Standard deviation threshold values for each use case.

Use case	Standard deviation threshold value (σ)
Testbed simulation	0.040
Spring coil	0.020
Door-line emblem	0.050
Child safety lock	0.075
Underbody	0.020
Wheel lock	0.040

ferent use case locations will exhibit appreciably different amounts of fluctuation in the correlation values. The fluctuations in turn affect the standard deviation values, and hence, the threshold of standard deviation has to be adjusted on a use case to use case basis to accurately detect occlusions. This new threshold information may be added into the existing 'usecase_info.txt' file that VI currently uses to keep track of configuration information for each use case.

3.3 Analysis of correlation values generated by the simulation on the testbed

The final simulation on the testbed was performed to test the reliability of the new algorithm, and the newly selected standard deviation threshold value in detecting occlusions on simulated data. A series of 20 inspections were performed, and the results were compared with the ground-truth established early on. As expected, the algorithm was able to successfully identify all brief and intermittent occlusions, based on the comparison of the observed standard deviation of the correlation values and the threshold standard deviation value selected previously. Inspections with no occlusions were also correctly reported - there were no false positives. Partial occlusions were not detected by this algorithm - there were four false negatives. The results are presented in table 3.3. The algorithm managed to detect 12 out of 16 occlusions, for a detection accuracy of 75%. The overall result for this

Table 3.3: Comparison of the output of the occlusion detection algorithm with the ground-truth information for the testbed simulation.

Minion no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/ no)
1	1	0	No
2	1	0	No
3	1	0	No
4	1	1	Yes
5	1	1	Yes
6	1	1	Yes
7	1	1	No
8	1	1	No
9	1	1	No
10	1	1	Yes
11	1	1	Yes
12	1	1	Yes
13	1	0	No
14	1	1	Yes
15	1	1	Yes
16	1	1	Yes
17	1	1	Yes
18	1	1	Yes
19	1	1	Yes
20	1	1	No

use case was 16 correct inspections out of 20, for an overall accuracy of 80%.

3.4 Analysis of correlation values generated by live data

3.4.1 Results for the spring coil use case

The correlation values from the live data for the spring coil use case was input to the new algorithm, along with the newly selected standard deviation threshold value for the spring coil use case. A series of 150 inspections were performed, and the results were compared with the ground-truth established early on. The algorithm was able to successfully identify most of the occlusions. Four false negative results happened because the inspection

Table 3.4: Truncated results of the occlusions detected on live data for the spring coil use case.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/ no)
1	1	1	Yes
1	2	0	No
2	1	1	No
2	2	1	Yes
3	1	0	No
3	2	0	No
4	1	0	No
4	2	0	No
5	1	1	No
5	2	1	No

area had a consistent appearance during inspection, even while being occluded. The results (truncated to show the first five cars) are presented in table 3.4. The algorithm managed to detect 22 out of 26 occlusions, for a detection accuracy of 84.6%. The overall result for this use case was 146 correct inspections out of 150, for an overall accuracy of 97.3%.

3.4.2 Results for the door-line emblem use case

The correlation values from the live data for the door-line emblem use case was input to the new algorithm, along with the newly selected standard deviation threshold value for the door-line emblem use case. A series of 154 inspections were performed, and the results were compared with the ground-truth established early on. The algorithm successfully detected all non-occlusions. The use case did not have any occlusions to detect, and the algorithm did not generate any false positives. The results (truncated to show the first five cars) are presented in table 3.5. The overall result for this use case was 154 correct inspections out of 154, for an overall accuracy of 100%.

Table 3.5: Truncated results of the occlusions detected on live data for the door-line emblem use case.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/ no)
1	1	0	No
1	2	0	No
2	1	0	No
2	2	0	No
3	1	0	No
3	2	0	No
4	1	0	No
4	2	0	No
5	1	0	No
5	2	0	No

3.4.3 Results for the child safety lock use case

The correlation values from the live data for the child safety lock use case was input to the new algorithm, along with the newly selected standard deviation threshold value for the child safety lock use case. A series of 125 inspections were performed, and the results were compared with the ground-truth established early on. The algorithm was able to successfully identify all of the inspections. The use case did not have any occlusions. No false positives were generated. The results (truncated to show the first five cars) are presented in table 3.6. The result for this use case was 125 correct inspections out of 125, for an overall accuracy of 100%.

3.4.4 Results for the underbody use case

The correlation values from the live data for the underbody use case was input to the new algorithm, along with the newly selected standard deviation threshold value for the underbody use case. A series of 264 inspections were performed, and the results were compared with the ground-truth established early on. The algorithm was able to successfully identify all of the occlusions. Interestingly, the live data had two cars for which

Table 3.6: Truncated results of the occlusions detected on live data for the child safety lock use case.

Car no.	Inspection area no.	Pass no. (no. of times triggered)	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/ no)
1	1	1	0	No
1	1	2	0	No
2	1	1	0	No
2	1	2	0	No
3	1	1	0	No
3	1	2	0	No
4	1	1	0	No
4	1	2	0	No
5	1	1	0	No
5	1	2	0	No

the inspection could only be performed on two of the four triggers because the trigger window was occluded for the complete duration of inspection. These 22 cases have not been included in the results. The results (truncated to show the only the first car) are presented in table 3.7. The algorithm managed to detect 60 out of 60 occlusions, for a detection accuracy of 100%. The overall result for this use case was 264 correct inspections out of 264, for an overall accuracy of 100%.

3.4.5 Results for the wheel lock use case

The correlation values from the live data for the wheel lock use case was input to the new algorithm, along with the newly selected standard deviation threshold value for the wheel lock use case. A series of 19 inspections were performed, and the results were compared with the ground-truth established early on. The algorithm successfully detected all non-occlusions. The use case did not have any occlusions to detect, and the algorithm did not generate any false positives. The live data also had 10 cars for which the inspection could not be performed at all because the trigger window was occluded for the complete duration of inspection. These 10 cases have not been included in the results. The results

Table 3.7: Truncated results of the occlusions detected on live data for the underbody use case.

Car no.	Trigger no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/no)
1	1	1	1	Yes
1	1	2	0	No
1	1	3	0	No
1	1	4	0	No
1	1	5	0	No
1	1	6	0	No
1	2	1	0	No
1	2	2	0	No
1	2	3	0	No
1	2	4	1	Yes
1	2	5	1	Yes
1	2	6	1	Yes
1	3	1	0	No
1	3	2	0	No
1	3	3	0	No
1	3	4	0	No
1	3	5	0	No
1	4	1	0	No
1	4	2	0	No
1	4	3	0	No
1	4	4	0	No
1	4	5	0	No

Table 3.8: Truncated results of the occlusions detected on live data for the wheel lock use case.

Car no.	Inspection area no.	Occlusion ground-truth (0:no; 1:yes)	Occlusion detected? (yes/ no)
1	1	0	No
2	1	0	No
3	1	0	No
4	1	0	No
5	1	0	No

(truncated to show the first five cars) are presented in table 3.8. The overall result for this use case was 19 correct inspections out of 19, for an overall accuracy of 100%.

3.5 Results of net occlusions detected

The results of the occlusions detected on the live data were collated, and the net occlusions were matched-up with the ground-truth established earlier. This is presented in table 3.9. Each row in the table represents the results for a single use case. Column four (No. of occlusions detected) indicates the performance of the new algorithm for each use case. Column five specifies the number of occlusions missed by the algorithm. For each use case, the sum of counts in columns four and five will match the number of occlusions in column three. Overall, the algorithm managed to detect 94 out of 102 occlusions. The total result for the algorithm across all the use cases was 732 correct inspections out of 740, for an occlusion detection accuracy of 98.9%.

Table 3.9: Comparison of the net occlusions detected vs. the ground-truth information.

Use case	Total inspections performed	Total no. of occlusions (ground-truth)	No. of occlusions detected	Undetected occlusions due to consistent appearance	No. of false positives
Testbed simulation	20	16	12	4	0
Spring coil	150	26	22	4	0
Door-line emblem	154	0	0	0	0
Underbody	264	60	60	0	0
Child safety lock	125	0	0	0	0
Wheel lock	19	0	0	0	0

Chapter 4

Conclusion and Future Work

In this thesis we have detailed one possible method by which occlusions may be detected within VI. We have modified VI to generate additional correlation information, and have analyzed this information to determine the variation in correlation values when occlusions happen. From this analysis, we have then proposed a fast and simple method that looks at the standard deviation of the correlation values to predict if an occlusion occurs. This new algorithm has demonstrated an occlusion detection accuracy of 98.9%. It works well to detect brief and intermittent occlusions - i.e., when there is a large fluctuation in the correlation values because of occlusion. Considering just the standard deviation of the correlation value does not work in cases where the part being inspected is partially occluded, because the correlation values remain more or less constant throughout the occlusion. Such cases are not handled by this new functionality. It also ignores cases where the occlusion covers the trigger window. In addition, we have observed that the algorithm is sensitive to a few parameters such as: shadows or change in intensity over inspection areas, vibrations or excessive shaking of the assembly during inspection, and the camera zoom level or distance between the camera and the subject of inspection. These parameters determine the level of fluctuation of the correlation values generated by VI itself, which in turn affect the standard deviation values used in analysis by the new algorithm.

For future work, we suggest that the functionality could be extended to detect partial occlusions. Additional metrics such as the duration of the trigger, the mean correlation values for the duration of the trigger, the difference in peak correlation values between classes, and the difference in mean correlation values across classes may potentially be utilized to detect partial occlusions.

Bibliography

[1] Jorg Schulte Adam W. Hoover. *Visual Inspector User Manual.* Clemson University, 2015.

[2] Xin Li Alper Yilmaz and Mubarak Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 26(11):1531–1536, 2004.

[3] M. Ortega B. Cancela and M. G. Pendo. Multiple human tracking system for unpredictable trajectories. *Machine Vision and Applications*, 25(2):511–527, 2014.

[4] Philip McLauchlan Benjamin Coifman, David Beymer and Jitendra Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, 6(4):271–288, 1998.

[5] Octavia Camps Camillo Gentile and Mario Sznaier. Segmentation for robust tracking in the presence of severe occlusion. *IEEE Transactions on Image Processing*, 13(2):13, 2004.

[6] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 70(1):41–54, 2006.

[7] Chung-Lin Huang and Wen-Chieh Liao. A vision-based vehicle identification system. *IEEE International Conference On Pattern Recognition*, 4(1):4, 2004.

[8] Shrinivas J. Pundlik Neeraj K. Kanhere and Stanley T. Birchfield. Vehicle segmentation and tracking from a low-angle off-axis camera. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2(2):6, 2005.

[9] Ichiro Ide Shuichi Sakai Okihisa Utsumi, Koichi Miura and Hidehiko Tanaka. An object detection method for describing soccer games from video. *IEEE International Conference OnMultimedia and Expo*, 1(1):45–48, 2002.

[10] Xuefeng Song and Ram Nevatia. A model-based vehicle segmentation method for tracking. *IEEE International Conference On Computer Vision*, 2(1):1124–1131, 2005.

[11] Xinge You Xin Li, Zhenyu Hu and C.L. Philip Chen. A novel joint tracker based on occlusion detection. *Knowledge-Based Systems*, 1(1):409–418, 2014.

[12] Xin Liu Yan Chen, Yingju Shen and Bineng Zhong. 3d object tracking via image sets and depth-based occlusion detection. *Signal Processing*, 1(1):146–153, 2014.