Automated Vision Inspection System for a Plastic Injection Mould Component

R Deepa¹, S Usha², P V Shashi Kumar³

Central Manufacturing Technology Institute, Tumkur Road, Bangalore-560022
¹E-mail: deepar@cmti-india.net
²E-mail: usha@cmti-india.net
³E-mail: pvs@cmti-india.net

Abstract

This paper presents an automated vision based defect inspection and sorting system for a plastic injection mould component called a Retractor Retaining Bush, which is an automotive safety critical component. The system identifies defects which usually occur in a plastic injection mould component. Various types of sensors and actuators were interfaced with the vision hardware and the part handling mechanism, to complete the total automated vision based inspection system.

Keywords: Vision Inspection, Plastic inspection, Defect identification

1 Introduction

The increased quality requirements and competitive market is necessitating the use of high speed and highly reliable systems for inspection. The improved quality control systems are mainly automated visual inspection (AVI) systems which replace human expertise to perform constant round-the-clock inspection. A human inspector is affected by fatigue and cannot maintain a high level of error trapping for long periods of time, can be slower than a machine and there will be inconsistency between different inspectors. Whereas, a vision system can work for long hours and has an average speed that is higher than that of a human being. The reliability of these systems is a very important factor, as these systems should guarantee that no critical defect goes undetected and also that no good components are detected as defective. Hence, a careful design and feasibility study must be made first, taking into account the advantages and disadvantages of the various options. In [1], the various advantages and disadvantages of visual inspection systems in industry are examined.

The AVI systems comprise of cameras, lightings, optics and the application software to make precise detection and classification. The application software is based on processing the images acquired with the camera and as such these automated visual inspection systems are built for a specific application.

AVI systems are used in a variety of application areas. The reference list indicates the wide range of application areas where AVI systems are implemented. AVI systems are used for checking of cracks, flaws, scratches and other defects in steel, plastic, tiles, fabric etc as discussed in [2-6], barcode identification, ensuring that lids and labels are properly applied to food and pharmaceutical products as discussed in [7-9], assembly inspection, grading of agricultural products such as seed corn or fruit as discussed in [10, 11], evaluating parts against their CAD data as discussed in [12] etc. In [13], a list of publications, reports and articles dealing with automated visual inspection for various industries is presented.

In [3], an automated visual inspection system for detection and classification of defects encountered in production of moulded plastic products was presented. Algorithms were developed for detection of shape defects like flash and short moulding and surface defects like jetting incarty and welding lines. Shape defects were identified with boundary tracking, calculation of pattern vectors followed by comparison with the prototype. Surface defects were identified with differential gradient operator and local gray-scale inhomogenity for line detection and spot detection respectively.

The core idea of this work was to develop an automated vision based defect sorting and inspection system for a plastic injection moulded component called a Retractor Retaining Bush (shown in Figure 1), which is an automotive component. The retaining bush is a component with complex shape and is produced at the rate of 2000/hr and currently the components are being inspected manually at the rate of 18000-20000 components/ day necessitating the use of such an automated system. The proposed work is challenging because of the complex shape of the component and the various types of defects which the component can encounter.

The main contributions of this paper are
Automated Vision Inspection System for a Plastic Injection Mould Component

- Development of software for identification and sorting of defect free components. This involves a training phase for training a master component and an inspection phase to run the inspection sequence.
- Development of GUI and interfacing of sensors to complete the automated system.

![Figure 1](image1.png)
(a) (b)  
Figure 1 Top (a) and Side View (b) of the component

A feeding system consisting of a vibratory feeder and a linear feeder was built to feed the components to the inspection station. The system built at CMTI, Bangalore is shown in Figure 2.

![Figure 2](image2.png)
Figure 2: The automated retractor retaining bush sorting system at CMTI, Bangalore

2 Hardware Details

The inspection station consists of two cameras (AVT Pike F-032B of resolution 640 x 480), one camera viewing the component from the top, and the other camera viewing the component approximately at right angle. This type of arrangement captures majority of the defects occurring in the component. The components are dumped into the bowl of a vibratory feeder system which then orients and feeds the component in a particular orientation through the linear feeder. The complete view of the component including the base can be provided by adding a third camera or modifying the feeding mechanism suitably. The arrival of the component at the station is sensed by an optical sensor, which in turn actuates a double acting cylinder to extend. This presents the component to the two cameras. The extension of the cylinder to the complete stroke is sensed by an inductive proximity sensor. After the inspection the controller signals the respective ejector to eject the component into the 'Accept' or 'Reject' bin based on the output from the vision application software. The cylinder retracts and the cycle continues.

The different types of defects commonly encountered in this component and to be inspected are:
1. Short shot which results in a partial part.
2. Component Discoloration.
3. Either of the holes in the component being filled.
4. Flash being present on the outer rim of the component.
5. Black spots on the component.
6. The part being deformed, like a bend in the lugs.

3 Defect Identification

Before the start of inspection, the component is trained to the system to take care of any environmental changes or lighting variations which may cause a change in the feature thresholds. The training methodology is explained in detail in the subsequent sections. The training phase stores the background images as seen by both the cameras and also extracts the thresholds of the features which define the different defect types.

During the inspection phase, the component, once it enters the inspection station is captured by both the top and the side cameras. The feature values are extracted and compared with the reference threshold values which were captured and stored during the training phase. The component is classified as defective if the feature value is out of range (Reference Value ± Tolerance Value). Dimensional measurements of two critical dimensions are provided for components which are defect free.

The HALCON image processing library was used to develop the application. Simple image processing techniques are followed to identify and classify the defects. The features selected to define the different defects and the methodologies used to identify them are as described below.

The component is extracted from both images by performing subtraction of the background which was saved in the training phase. This extracts the component by eliminating the background and other external disturbances in both the top and side camera images.

Table 1 indicates the defects which are identified by the top and side cameras, the variable names given to the feature thresholds and the flag which is set for the respective defect type.

Initially, before the start of inspection, all flags are set to zero. Identification of any one of the defects will cause the component to be rejected without checking for the presence of other defects, since the defect checking happens sequentially. Variables are shown in italics and underlined.
### Table 1: Variable names given to the feature thresholds and the flag which is set for the respective defect type

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Camera</th>
<th>Defect Type</th>
<th>Feature Threshold variable</th>
<th>Flag which defines the defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Top Camera</td>
<td>Short shot (Broken Component)</td>
<td>AreaBroComp</td>
<td>BrokenComponentFlag1</td>
</tr>
<tr>
<td>2</td>
<td>Top Camera</td>
<td>Component Discoloration</td>
<td>AreaColor1</td>
<td>ColorChangeFlag1</td>
</tr>
<tr>
<td>3</td>
<td>Top Camera</td>
<td>Filled Hole</td>
<td>MinHoleArea, MaxHoleArea, Distance</td>
<td>FilledHoleFlag1</td>
</tr>
<tr>
<td>4</td>
<td>Top Camera</td>
<td>Flash</td>
<td>AreaOuter</td>
<td>FlashFlag1</td>
</tr>
<tr>
<td>5</td>
<td>Top Camera</td>
<td>Black Spots</td>
<td>SpotsThreshCAM1</td>
<td>SpotsFlag1</td>
</tr>
<tr>
<td>6</td>
<td>Side Camera</td>
<td>Component Discoloration</td>
<td>AreaColor2</td>
<td>ColorChangeFlag2</td>
</tr>
<tr>
<td>7</td>
<td>Side Camera</td>
<td>Short shot</td>
<td></td>
<td>MissingLugFlag</td>
</tr>
<tr>
<td>8</td>
<td>Side Camera</td>
<td>Bent Lug</td>
<td></td>
<td>BentLugFlag</td>
</tr>
<tr>
<td>9</td>
<td>Side Camera</td>
<td>Black Spots</td>
<td>SpotsThreshCAM2</td>
<td>SpotsFlag2</td>
</tr>
</tbody>
</table>

#### 3.1 Broken Component

**Teaching:** Area of the master component (Number of pixels) is found and stored in the variable AreaBroComp.

**Inspection:** Area of the test component is found. If Area < (AreaBroComp - 2000), then the flag BrokenComponentFlag1 is enabled.

![Figure 3: Broken component](image)

#### 3.2 Color Change

**Teaching:** Linear smoothing of the master component image is performed using a mean filter. The area of the component is found considering the gray values in the region of the mean filtered master component image and stored in the variable AreaColor1.

\[
\text{AreaColor1} = \sum_{(r,c)} g(r,c)
\]

Where, \( g(r,c) \) is the gray value function in the region of the mean filtered master component image.

**Inspection:** Mean filter of the test component image is found using a mask size of 11 x 11. The area of the component is found considering the gray values in the region of the mean filtered test component image and stored in the variable AreaGray.

\[
\text{AreaGray} = \sum_{(r,c)} f(r,c)
\]

Where, \( f(r,c) \) is the gray value function in the region of the mean filtered test component image.

If AreaGray < (94% of AreaColor1), then the flag ColorChangeFlag1 is enabled. The same methodology is followed for identifying color change in the side camera view.

#### 3.3 Filled Hole

**Training:** The boundary of the master component is extracted using edge detection by ‘lanser’ method applied on the mean filtered image of the component. Lanser edge detection method is the isotropic version of the deriche’s edge detection method. Deriche used Canny’s approach to find optimal edge filters, that can be implemented recursively. Canny’s formulation indicates that the Gaussian filter is the optimal smoothing filter. But since the Gaussian filter and its derivatives cannot be implemented recursively, the execution time depends on the amount of smoothing. The derivative filters can be implemented recursively but they are anisotropic i.e., the edge amplitude depends on the angle of the edge in the image. This is undesirable because it makes the selection of relevant edges harder.

The area of the boundary extracted using ‘lanser’ method is computed and stored in the variable MaxArea. Further, Regions with area between MaxArea 5.5 and MaxArea 11.0 are selected. This
selects the two holes. The areas of the two holes are found, minimum of the two is stored in \( \text{MinHoleArea} \) and Maximum of the two is stored in \( \text{MaxHoleArea} \). Distance between the centers of the two holes is computed and stored in the variable \( \text{Distance} \).

**Inspection:** The boundary of the test component is extracted using edge detection by ‘lanser’ method applied on the mean filtered image of the test component. Further, regions with area between \( (\text{MinHoleArea} - 100) \) & \( (\text{MaxHoleArea} + 100) \) are selected. This selects the holes. If number of holes is less than 2 or distance between centers of the holes is less than \( (0.75 \times \text{Distance}) \), then the flag \( \text{FilledHoleFlag1} \) is enabled.

![Figure 4: Component with one of the holes filled](image)

**3.4 Flash**

**Training:** The flash normally occurs at the outer rim of the component as indicated by the rectangle in the figure 5. This outer rim of the master component is extracted using edge detection by ‘lanser’ method applied on the mean filtered image of the master component and the area of this outer rim is computed and stored in the variable \( \text{AreaOuter} \).

![Figure 5: Flash at the outer rim of the component being identified](image)

**Inspection:** Three conditions are checked to find the presence of flash on the outer rim of the test component. If either of the condition is true, the presence of flash is identified and the component is rejected. Firstly, the outer rim of the component is extracted using edge detection by ‘lanser’ method applied on the mean filtered image of the test component. The area of the outer rim is computed and stored in the variable \( \text{Area} \). If \( \text{Area} > (\text{AreaOuter} + 75) \) or \( \text{Area} < (\text{AreaOuter} - 40) \), then the flag \( \text{FlashFlag1} \) is enabled. This method identifies the flash whose shape causes discontinuities or elongation of the detected edge. Secondly, an ellipse is fit to the detected edge and regions which are protruding out from the fitted ellipse and having an area above 35 pixels are selected. The threshold of 35 pixels is set to avoid false detections. If this method returns any region, then the flag \( \text{FlashFlag1} \) is enabled. Thirdly, the extracted edge contour is segmented at dominant points. Dominant points are points at which there is a prominent change in direction of the contour. If the edge contour is smooth without discontinuities caused by the presence of flash, then the contour will remain un-segmented.

Presence of flash causes the segmentation of the contour. If number of segments after applying segmentation is greater than one, then the flag \( \text{FlashFlag1} \) is enabled.

3.5 **Black Spots**

**Training:** The Median Filter is applied on the master component image to remove noise which could be categorized wrongly as a black spot. Dark circular dots are extracted from this median filtered image by using a dot filter having a circular filter mask of size 13. Dots with area between 30 and 300 and convexity between 0.8 and 1 are selected from this dot filtered image. This step will extract multiple spots, which are nothing but lighting variations on the surface of the component. The gray value with the highest frequency of occurrence is computed in each of the dots, from their histograms. The minimum out of these gray values is stored in the variable \( \text{SpotsThreshCAM1} \). This threshold will be used in the inspection phase to distinguish between spots occurring due to lighting variations and the actual black spots caused in the injection moulding process.

**Inspection:** The test component image is Median Filtered to remove noise. Dark circular dots are extracted from this median filtered image by using a dot filter having a circular filter mask of size 13. Dots with area between 23 and 285 and convexity between 0.6 and 0.99 are extracted. The gray value with the highest frequency of occurrence is computed in each of these dots, from their histograms. Dots which have their highest occurring gray value less than \( \text{SpotsThreshCAM1} \) is categorized as black spots occurring due to injection moulding. If the number of such dots is greater than 0, then the flag \( \text{SpotsFlag1} \) is enabled. The same methodology is followed for identifying black spots in the side camera view. Figure 6 shows the black spots being identified in both the views.

![Figure 6: Black spots on the component being identified in top view (a) and side view (b)](image)
3.6 Missing Lug

**Inspection:** The lug regions in the test component image are extracted within the rectangle as shown in figure 7. Bounding box is fitted to the lugs and the leftmost positions of the bounding box are extracted to find half broken lugs. If the number of lugs is less than 2 or if the length difference between the lugs is greater than 50 pixels, then the flag *MissingLugFlag* is enabled.

![Figure 7: Missing Lug of the component being identified](image)

3.7 Bent Lug

**Inspection:** The lug regions in the test component image are extracted within the rectangle as shown in figure 8. Bounding box is fitted to the lugs and the orientation of the bounding boxes is used to extract the orientation of the lugs. If the difference in orientation between the bounding boxes of the two lugs is more than 5°, then the flag *BentLugFlag* is enabled.

![Figure 8: Bent Lug of the component being identified](image)

4 Automation and Graphical User Interface (GUI)

Figure 9 shows the schematic of the automated vision based inspection system.

![Figure 9: Schematic of the automated vision based inspection system](image)

Apart from the vision based algorithm development, interfacing of different sensors and actuators was done to complete the automation of the system. An optical slot sensor, an inductive proximity sensor, a solenoid valve and two relays for the two ejectors is interfaced in the application through an Advantech PCI-1761DIO card.

Figure 10 shows the GUI of the application during the training and inspection phases. GUI and interfacing is done on the Visual Studio development platform.

![Figure 10(a): Screen1 to choose the training or inspection phase](image)

**Figure 10(b): Training Window which has controls for capturing the background, changing the threshold for selecting the component and training the system with the features of the master component**

![Figure 10(c): Inspection Window to run the inspection system automatically](image)

5 Results and Discussion

About 250 good samples and 250 defective samples were tested to ascertain the reliability and validity of the developed algorithms to variations in the nature of defects and its effectiveness in the factory environment where there will be environmental changes which could affect the performance. The performance of the defect identification system is as indicated in table 2.
6 Conclusion

This paper presents an automated vision based defect inspection and sorting system for the retaining bush. The retaining bush is a plastic injection mould component which develops various types of defects during its production. This system is able to capture defects like partial part, component discoloration, burn marks, flash etc, which generally occur in an injection mould component. System can be tuned to include other surface defects like porosity which occur due to injection moulding. Algorithms for detection and sorting of defective components and measurement of the critical dimensions were developed. A vibratory feeder and linear feeder were integrated with the vision system for part presentation. A Graphical User Interface was developed for operating the system. Various types of sensors and actuators were interfaced with the vision hardware and the part handling mechanism to complete the total automated vision based inspection system for defect inspection and sorting of the retaining bush.

References


Table 2: Results of the Defect Identification System

<table>
<thead>
<tr>
<th>Good Components</th>
<th>Defective components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classified as good components</td>
<td>90%</td>
</tr>
<tr>
<td>Classified as bad components</td>
<td>10%</td>
</tr>
</tbody>
</table>