A COMPARATIVE STUDY OF MACHINE VISION CLASSIFICATION
TECHNIQUES FOR THE DETECTION OF MISSING CLIPS

By

Brandon Miles

A thesis submitted to the Department of Mechanical and Materials Engineering in conformity with the requirements for the degree of

Master of Science (Engineering)

Queen’s University
Kingston, Ontario, Canada
July 2009

Copyright © Brandon Miles, 2009
Abstract

This thesis provides a comparative study of machine vision (MV) classification techniques for the detection of missing clips on an automotive part known as a cross car beam. This is a difficult application for an automated MV system because the inspection is conducted in an open manufacturing environment with variable lighting conditions.

A laboratory test cell was first used to investigate the effect of lighting. QVision, a software program originally developed at Queen’s University, was used to perform a representative inspection task. Solutions with different light sources and camera settings were investigated in order to determine the best possible set up to acquire an image of the part. Feature selection was applied to improve the results of this classification.

The MV system was then installed on an industrial assembly line. QVision was modified to detect the presence or absence of four clips and communicate this information to the computer controlling the manufacturing cell. Features were extracted from the image and then a neuro fuzzy (ANFIS) system was trained to perform the inspection. A performance goal of 0% False Positives and less than 2% False Negatives was achieved with the feature based ANFIS classifier. In addition, the problem of a rusty clip was examined and a radial hole algorithm was used to improve performance in this case. In this case, the system required hours to train.

Five new classifiers were then compared to the original feature based ANFIS classifier: 1) feature based with a Neural Network, 2) feature based with principle component analysis (PCA) applied and ANFIS, 3) feature based with PCA applied and a Neural Network, 4) Eigenimage based with ANFIS and 5) Eigenimage based with a Neural Network. The effect of adding a Hough rectangle feature and a principle component colour feature was also studied. It was found that the Neural Network classifier performed better than the ANFIS classifier. When PCA was applied the results improved still further. Overall, feature based classifiers had better performance than Eigenimage based classifiers. Finally, it should be noted that these six classifiers required only minutes to train.
Acknowledgements

I would like to thank Professor Surgenor for his supervision of this project. He has provided an incredible opportunity to learn. He has also been an excellent supervisor providing support and opportunities throughout the course of my studies here. Thank you.

I would also like to thank the people at Van Rob for all their time and assistance on this project. Arvin Ocampo, Bharat Sharma, Cheng Zeng, Sam Li, the operators and all the other people that helped me at the plant have been incredibly supportive through this endeavour and without them this work would not have been possible.

The financial support of AUTO 21 and OCE have made this project possible along with Queen’s University and the department of Mechanical of Materials Engineering.

I would also like to thank my family who have encouraged and supported me in the past 2 years. My Mom and Dad: Tony and Janice, along with my sisters Laura and Kate and Grandpa are an incredible blessing in my life. You guys have been a constant support in my studies and in my life, helping me to grow in many ways including knowing God more.

Alex, Ben B., Ben N., Brenda, Cliff, Chris, Flora, Gilbert, Scott and Steve you guys are incredible friends. Thank you for your support and encouragement.
# Table of Contents

**ABSTRACT**  
**ACKNOWLEDGEMENTS**  
**TABLE OF CONTENTS**  
**LIST OF FIGURES**  
**LIST OF TABLES**  
**NOMENCLATURE**  

**CHAPTER 1: INTRODUCTION**  
1.1 Problem Overview  
1.2 Thesis Objectives  
1.3 Organization of Thesis  

**CHAPTER 2: BACKGROUND AND LITERATURE REVIEW**  
2.1 Machine Vision for Inspection  
2.2 Lighting the Object  
2.3 Feature Based Methods  
  2.3.1 Hough Transform  
  2.3.2 Corner Detectors  
  2.3.3 Colour Based Features  
  2.3.4 Textures  
  2.3.5 SIFT  
  2.3.6 Template Matching  
2.4 Principle Component Analysis  
2.5 Processing Methods  
  2.5.1 Neural Networks  
  2.5.2 Fuzzy Logic  
  2.5.3 Neuro Fuzzy Processing  
  2.5.4 Feature Selection  
2.6 Receiver Operating Characteristic Graphs  
2.7 Previous Work  
  2.7.1 Clip Detection  
  2.7.2 Fuzzy Logic  
  2.7.3 Neural Networks  
  2.7.4 ANFIS  
  2.7.5 PCA  
2.8 Summary  

iv
APPENDIX A: Camera and Lighting Specifications 141
   A.1 Camera Specifications 141
   A.2 Lighting Specifications 141

APPENDIX B: Dataset Description 142
   B.1 Features used for ‘No Rust Dataset’ in Chapter 4 142
   B.2 Features used for ‘Original’ dataset in Chapter 5 148
   B.3 Features used for ‘New’ dataset in Chapter 5 153
   B.4 Features used for ‘Orientation’ dataset in Chapter 5 158

APPENDIX C: Algorithm Details and Code Listings 163
   C.1 Summary of Algorithms 163
   C.2 Feature Extraction Algorithms 163
      C.2.1 Geometric and Colour Features 163
      C.2.2 Regions of Interest 173
   C.3 Classifier Algorithms 174
      C.3.1 ANFIS 174
      C.3.2 Neural Network 175
      C.3.3 PCA ANFIS 176
      C.3.4 PCA Neural Network 176
      C.3.5 Eigenimage ANFIS 177
      C.3.6 Eigenimage Neural Network 177
   C.4 Feature Selection Routines 178
      C.4.1 Exhaustive Method 178
      C.4.2 Forward Method 179
      C.4.3 Forward (many) Method 179
      C.4.4 Exhaustive with Prefiltering Method 180
   C.5 Performance Evaluation 181
      C.5.1 RMS 181
      C.5.2 ROC 181
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>2</td>
<td>Original cross car beam with J-clips and push nuts inserted.</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>3</td>
<td>A beam being manually inspected for the presence of fasteners.</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>8</td>
<td>Lighting techniques.</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>10</td>
<td>The Hough Transform process; a) region of interest from input image, b) image converted to greyscale, c) Image of the gradient Hough transform of the grey scale image. The location of the line (the brightest peak) is circled in red and d) the line found by the Hough transform highlighted on the image of the clip.</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>13</td>
<td>HSV circle with Hue and Saturation components shown.</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>16</td>
<td>left image = Dataset before applying PCA, right image = after applying PCA.</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>20</td>
<td>Sample Neural Network with two layers.</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>22</td>
<td>left – Gaussian Membership Function c = 50, σ = 20, right – Bell Shaped Membership Function a = 20, b = 4, c= 50.</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>23</td>
<td>Diagram of Fuzzy Reasoning for a Single Input Single Rule System with the rule if $x$ is $A$ then $y$ is $B$ on the left and if $x$ is $A'$ then $y$ is $B'$ on the right.</td>
</tr>
<tr>
<td>Figure 2.8</td>
<td>25</td>
<td>Structure of a 1 input 3 rule Sugeno FIS system.</td>
</tr>
<tr>
<td>Figure 2.9</td>
<td>26</td>
<td>Sample Sugeno-Type ANFIS System (with layers 2 &amp; 3 combined).</td>
</tr>
<tr>
<td>Figure 2.10</td>
<td>31</td>
<td>Sample ROC graphs left image = poor classification results, right image = excellent classification results.</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>41</td>
<td>Manual inspection of cross car beam.</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>42</td>
<td>A radio bracket with clips installed.</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>44</td>
<td>Image of Configuration Module.</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>45</td>
<td>Illustration of circular gradient detection method.</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>47</td>
<td>Sample input data to the system. The blue upside down triangle represents the data point for a pass image and the red triangle represent an input for a fail image.</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>48</td>
<td>Sample membership function. The blue and red triangles represent pass and fail images respectively. The curves represent the membership function. The degree of membership in an MF for any given point is between 0 and 1.</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>49</td>
<td>Sample Network for ANFIS system.</td>
</tr>
<tr>
<td>Figure 3.8</td>
<td>50</td>
<td>Output results of training. Green dots represent correctly classified images.</td>
</tr>
<tr>
<td>Figure 3.9</td>
<td>51</td>
<td>Various optimizations for the data set. The solid circles denote the features used.</td>
</tr>
<tr>
<td>Figure 3.10</td>
<td>52</td>
<td>The Camera Module for evaluating images as pass or fail.</td>
</tr>
<tr>
<td>Figure 3.11</td>
<td>53</td>
<td>Image of test cell.</td>
</tr>
<tr>
<td>Figure 3.12</td>
<td>53</td>
<td>Layout of the test cell, showing the three different lighting solutions.</td>
</tr>
<tr>
<td>Figure 3.13</td>
<td>54</td>
<td>Image with 3 different sources of light: left – Fluorescent, middle – LED spotlight, right – LED ring.</td>
</tr>
<tr>
<td>Figure 3.14</td>
<td>56</td>
<td>Ring light on, room lights on, camera auto off (pass image on left, failed on right).</td>
</tr>
<tr>
<td>Figure 3.15</td>
<td>56</td>
<td>Ring light on, room lights off, camera auto off (pass image on left, failed on right). Figure 3.16: Ring light on, room lights on, camera auto on (pass image on left, failed on right).</td>
</tr>
<tr>
<td>Figure 3.16</td>
<td>56</td>
<td>Ring light on, room lights on, camera auto on (pass image on left, failed on right).</td>
</tr>
<tr>
<td>Figure 3.17</td>
<td>56</td>
<td>Ring light on, room lights off, camera auto on (pass image on left, failed on right).</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>-----------------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure 3.18</td>
<td>60</td>
<td>Network structure for ANFIS system without feature selection.</td>
</tr>
<tr>
<td>Figure 3.19</td>
<td>60</td>
<td>Testing results without feature selection.</td>
</tr>
<tr>
<td>Figure 3.20</td>
<td>61</td>
<td>Network structure for ANFIS system with feature selection.</td>
</tr>
<tr>
<td>Figure 3.21</td>
<td>61</td>
<td>Testing results with feature selection.</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>64</td>
<td>The cross car beam for the new test cell.</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>66</td>
<td>Configuration screen. The user has the ability to select feature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parameters for the four clip positions.</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>66</td>
<td>Configuration screen showing features being defined on the bottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bracket.</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>67</td>
<td>Training Module updated to work with four clips.</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>67</td>
<td>Camera Module configured to work with four clips.</td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>69</td>
<td>a) An illustration of 4 connect pixels. The red and blue objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>are not connected. b) An illustration of 8 connected pixels.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>There is just one connected object.</td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>70</td>
<td>Illustration of Hough semi circle algorithm.</td>
</tr>
<tr>
<td>Figure 4.8</td>
<td>71</td>
<td>Illustration of the centroid algorithm for grey scale image, with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the ‘+’ sign denoting the centroid of the image.</td>
</tr>
<tr>
<td>Figure 4.9</td>
<td>71</td>
<td>Illustration of the Centroid algorithm. This is for a thresholded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>black and white image with the black ‘+’ sign denoting the centroid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of the image.</td>
</tr>
<tr>
<td>Figure 4.10</td>
<td>73</td>
<td>An illustration of the radial hole statistics feature.</td>
</tr>
<tr>
<td>Figure 4.11</td>
<td>74</td>
<td>The Testing Module.</td>
</tr>
<tr>
<td>Figure 4.12</td>
<td>75</td>
<td>The log file graphical display.</td>
</tr>
<tr>
<td>Figure 4.13</td>
<td>76</td>
<td>The camera setup screen for properly aligning the cameras and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>adjusting focus.</td>
</tr>
</tbody>
</table>
Figure 4.14 77 The PLC running the clipping cell next to PC running the vision system.

Figure 4.15 78 New clipping cell with the vision system located at 5.

Figure 4.16 79 a) Camera configuration (each camera images two clips), b) The bracket in the clipping cell.

Figure 4.17 80 left – Top Pass image, right – Top Fail image, from the manufacturing cell.

Figure 4.18 80 left – Bottom Pass image, right – Bottom Fail image, from the manufacturing cell.

Figure 4.19 84 The four available feature selection methods and their tuneable parameters.

Figure 4.20 85 One typical workflow for completing feature selection.

Figure 4.21 86 Two pass clips, for top bracket.

Figure 4.22 86 Two failed clips, for top bracket.

Figure 4.23 86 Two failed clips with rust present, top bracket.

Figure 4.24 89 Clip 3 No Rust results with no feature selection, presented in a scatter plot.

Figure 4.25 89 Clip 3 No Rust results with no feature selection, presented in a ROC graph.

Figure 4.26 90 Clip 3 No Rust results with feature selection, presented in a scatter plot.

Figure 4.27 90 Clip 3 No Rust results with feature selection, presented in a ROC graph.

Figure 4.28 92 Illustration of features used for clip 2 in Rust + New Algorithm.

Figure 4.29 94 Flow chart detailing acceptance process.

Figure 5.1 98 Network structure used to train system.
<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 5.2</td>
<td>99</td>
<td>Principal components used to train a Neural Network for the PCA Neural Network classifier. No scales have been provided because the degree of separation is the item of importance in this case.</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>100</td>
<td>ANFIS for network with 5 principle components as inputs.</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>102</td>
<td>Eigenimages, of the top left clip. These are greyscale images, although the maximum and minimum values may have been truncated to make them visible to the reader.</td>
</tr>
<tr>
<td>Figure 5.5</td>
<td>103</td>
<td>Sample coefficients obtain from Eigenimage decomposition of sample input images. These coefficients correspond to the Eigenimages in Figure 5.4. The upside down triangles represent pass images. The fail triangles represent fail images. The most important feature in this graph is the separation between pass inputs and fail inputs. This is why the horizontal scale has been omitted. The coefficients are used as inputs in the ANFIS and NN classifiers.</td>
</tr>
<tr>
<td>Figure 5.6</td>
<td>104</td>
<td>a) Pass image used for Eigenimage decomposition. b) The same image, with edge detection performed.</td>
</tr>
<tr>
<td>Figure 5.7</td>
<td>104</td>
<td>a) Fail image used for Eigenimage decomposition. b) The same image, with edge detection performed.</td>
</tr>
<tr>
<td>Figure 5.8</td>
<td>106</td>
<td>An illustration of the Hough rectangle method. Lines are voted for in two planes. The edges of the grey rectangle are found in both planes. In the one on the left none of the edges will overlap because the orientation is wrong. In the one on the right two sets of edges overlap resulting in a strong identification of the major and minor axis for the rectangle.</td>
</tr>
<tr>
<td>Figure 5.9</td>
<td>106</td>
<td>The Hough rectangle routine locating a clip in the image.</td>
</tr>
<tr>
<td>Figure 5.10</td>
<td>107</td>
<td>The Red, Green and Blue components of a Bracket.</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure 5.11</td>
<td>108</td>
<td>The three principal colour components of the bracket. 1) The bracket, 2) the principal colour component with the most variance, 3) the principal colour component with the second most variance, 4) the principal colour component with the least variance, 5) image 3 contrast increased by 31% and brightness by 100% to make the colour components more visible to the user, 6) image 4 contrast increased by 31% and brightness by 100%.</td>
</tr>
<tr>
<td>Figure 5.12</td>
<td>110</td>
<td>Input Images.</td>
</tr>
<tr>
<td>Figure 5.13</td>
<td>110</td>
<td>1) Median corner detector results pass image, 2) median corner detector results fail image, 3) median corner detector results pass image (brightness artificially increased by 75% to be visible to the reader), 4) median corner detector results fail image (brightness artificially increased by 75% to be visible to the reader).</td>
</tr>
<tr>
<td>Figure 5.14</td>
<td>111</td>
<td>Harris corner detector (image scaled and converted to binary to retain all the corner points).</td>
</tr>
<tr>
<td>Figure 5.15</td>
<td>111</td>
<td>Eigenvalue corner detector (image scaled and converted to binary to retain all the corner points).</td>
</tr>
<tr>
<td>Figure 5.16</td>
<td>113</td>
<td>Sample pass clip for the orientation image set.</td>
</tr>
<tr>
<td>Figure 5.17</td>
<td>113</td>
<td>Sample fail (missing clip) for the orientation image set.</td>
</tr>
<tr>
<td>Figure 5.18</td>
<td>114</td>
<td>Sample fail (backwards and upside down clip) for the orientation image set.</td>
</tr>
<tr>
<td>Figure 5.19</td>
<td>114</td>
<td>Sample fail (backwards) for the orientation image set.</td>
</tr>
<tr>
<td>Figure 5.20</td>
<td>114</td>
<td>Sample fail (upside down) for the orientation image set.</td>
</tr>
<tr>
<td>Figure 5.21</td>
<td>115</td>
<td>The region of interest for the top left clip found relative to the centre of the beam.</td>
</tr>
<tr>
<td>Figure 5.22</td>
<td>117</td>
<td>The features used in the first training set.</td>
</tr>
<tr>
<td>Figure B.1</td>
<td>144</td>
<td>Top left pass clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure B.2</td>
<td>144</td>
<td>Top left fail clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.3</td>
<td>145</td>
<td>Top right pass clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.4</td>
<td>145</td>
<td>Top right fail clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.5</td>
<td>146</td>
<td>Bottom left pass clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.6</td>
<td>146</td>
<td>Bottom left fail clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.7</td>
<td>147</td>
<td>Bottom right pass clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.8</td>
<td>147</td>
<td>Bottom right fail clip. Features used in ‘No Rust’ dataset in Chapter 4.</td>
</tr>
<tr>
<td>Figure B.9</td>
<td>149</td>
<td>Top left pass clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.10</td>
<td>149</td>
<td>Top left fail clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.11</td>
<td>150</td>
<td>Top right pass clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.12</td>
<td>150</td>
<td>Top right fail clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.13</td>
<td>151</td>
<td>Bottom left pass clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.14</td>
<td>151</td>
<td>Bottom left fail clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.15</td>
<td>152</td>
<td>Bottom right pass clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.16</td>
<td>152</td>
<td>Bottom right fail clip. Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure B.17</td>
<td>154</td>
<td>Top left pass clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.18</td>
<td>154</td>
<td>Top left fail clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.19</td>
<td>155</td>
<td>Top right pass clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.20</td>
<td>155</td>
<td>Top right fail clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.21</td>
<td>156</td>
<td>Bottom left pass clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.22</td>
<td>156</td>
<td>Bottom left fail clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.23</td>
<td>157</td>
<td>Bottom right pass clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.24</td>
<td>157</td>
<td>Bottom right fail clip. Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Figure B.25</td>
<td>159</td>
<td>Top left pass clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.26</td>
<td>159</td>
<td>Top left fail clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.27</td>
<td>160</td>
<td>Top right pass clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.28</td>
<td>160</td>
<td>Top right fail clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.29</td>
<td>161</td>
<td>Bottom left pass clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.30</td>
<td>161</td>
<td>Bottom left fail clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure B.31</td>
<td>162</td>
<td>Bottom right pass clip. Features used in ‘Orientation’ dataset in Chapter 5</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure B.32</td>
<td>162</td>
<td>Bottom right fail clip. Features used in ‘Orientation’ dataset in Chapter 5.</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1</td>
<td>58</td>
<td>Summarized performance test results for the four trials.</td>
</tr>
<tr>
<td>Table 4.1</td>
<td>87</td>
<td>Comparison of feature selection 1 and 2 (52 feature attributes versus 4 feature attributes).</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>91</td>
<td>Results with and without rusty clips.</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>92</td>
<td>Features used in <em>Rust + New Algorithm</em>.</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>97</td>
<td>Summary of the six classifiers.</td>
</tr>
<tr>
<td>Table 5.2</td>
<td>116</td>
<td>Results of edge detection on Eigenimage based classification on the original dataset.</td>
</tr>
<tr>
<td>Table 5.3</td>
<td>118</td>
<td>Results for the classifiers based on the original dataset.</td>
</tr>
<tr>
<td>Table 5.4</td>
<td>120</td>
<td>Testing results for extended testing set, trained on the original dataset and tested on the extended data set.</td>
</tr>
<tr>
<td>Table 5.5</td>
<td>121</td>
<td>Results of Hough rectangle without edge detection on original dataset.</td>
</tr>
<tr>
<td>Table 5.6</td>
<td>122</td>
<td>Results for PCA colour algorithm on original dataset.</td>
</tr>
<tr>
<td>Table 5.7</td>
<td>123</td>
<td>Results for new feature set on the original test set.</td>
</tr>
<tr>
<td>Table 5.8</td>
<td>124</td>
<td>Extended results trained on the original data with the new features and tested on the extended testing set.</td>
</tr>
<tr>
<td>Table 5.9</td>
<td>125</td>
<td>Results for feature based approach on orientation test set.</td>
</tr>
<tr>
<td>Table 5.10</td>
<td>126</td>
<td>Results for Eigenimage approach on orientation test set.</td>
</tr>
<tr>
<td>Table B.1</td>
<td>142</td>
<td>Features used in Feature Selection 2 from Table 4.1</td>
</tr>
<tr>
<td>Table B.2</td>
<td>142</td>
<td>Features used in ‘No Rust’ from Table 4.2</td>
</tr>
<tr>
<td>Table B.3</td>
<td>143</td>
<td>Features used in ‘NoRust unoptimised’ dataset and in Feature Selection 1 from Table 4.1 in Chapter 4.</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td>Text</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Table B.4</td>
<td>148</td>
<td>Features used in ‘Original’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Table B.5</td>
<td>153</td>
<td>Features used in ‘New’ dataset in Chapter 5.</td>
</tr>
<tr>
<td>Table B.6</td>
<td>158</td>
<td>Features used in ‘Orientation’ dataset in Chapter 5.</td>
</tr>
</tbody>
</table>
Nomenclature

Variables

\[ B_L \] lower bound on the classifier output
\[ B_U \] upper bound on the classifier output
\[ e_1, e_2, \ldots, e_n \] Eigenvectors of Eigenimages
\[ E_{rms} \] root mean squared (RMS) error
\[ I_{ring} \] intensity value of a given ring for radial hole statistics
\[ p_1, p_2, \ldots, p_n \] projection coefficients of image onto Eigenimage space
\[ r_{inner} \] radius of the inner ring for radial hole statistics
\[ r_{mid} \] radius of the middle ring for radial hole statistics
\[ r_{outer} \] radius of the outer ring for radial hole statistics
\[ r_w \] ring width for radial hole statistics
\[ S \] sample covariance matrix of a given input
\[ t_{thld} \] classification threshold
\[ w_1, w_2, \ldots, w_n \] firing strengths of fuzzy rules
\[ Z_i \] output of the classifier algorithm
\[ Z_i^d \] desired (correct) classification for the \( i^{th} \) image

Greek Variables

\[ \theta \] angle normal to the direction of a line in the Hough transform.
\[ \rho \] other position component of a line in the Hough transform (see \( \theta \))
\[ \tau \] strength of a line found by the Hough transform
\[ \mu_1, \mu_2 \] first and second eigenvalues in the Hessian matrix for corner detection
\[ \mu_A, \mu_B, \ldots, \mu_X \] fuzzy membership functions
\[ \Sigma \] covariance matrix of a given input
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANFIS</td>
<td>Adaptive Neuro Fuzzy Inferencing System</td>
</tr>
<tr>
<td>AVI</td>
<td>Automated Visual Inspection</td>
</tr>
<tr>
<td>DoG</td>
<td>Difference of Gaussians</td>
</tr>
<tr>
<td>FIS</td>
<td>Fuzzy Inferencing System</td>
</tr>
<tr>
<td>FN</td>
<td>False Negative</td>
</tr>
<tr>
<td>FP</td>
<td>False Positive</td>
</tr>
<tr>
<td>GHT</td>
<td>Generalised Hough Transform</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HSV</td>
<td>Hue Saturation Value colour space</td>
</tr>
<tr>
<td>MF</td>
<td>Membership Function</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi Layer Perceptron (Neural Network)</td>
</tr>
<tr>
<td>MV</td>
<td>Machine Vision</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
<tr>
<td>PCA</td>
<td>Principle Component Analysis</td>
</tr>
<tr>
<td>PLC</td>
<td>Programmable Logic Controller</td>
</tr>
<tr>
<td>RGB</td>
<td>Red Green Blue colour space</td>
</tr>
<tr>
<td>RMS</td>
<td>Root Mean Square</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristics</td>
</tr>
<tr>
<td>SFS</td>
<td>Sequential Forward Selection</td>
</tr>
<tr>
<td>SIFT</td>
<td>Scale Invariant Feature Transform</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
<tr>
<td><strong>Glossary</strong></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>backpropagation</td>
<td>A process by which weights or values in a network are updating by moving backwards from right to left based on the output error of the network.</td>
</tr>
<tr>
<td>crisp</td>
<td>A non fuzzy input (for example 2, 3, or 17 are crisp inputs).</td>
</tr>
<tr>
<td>defuzzification</td>
<td>The process of going from fuzzy values to crisp values.</td>
</tr>
<tr>
<td>Eigenimage</td>
<td>An image that can be used as a basis to describe other images.</td>
</tr>
<tr>
<td>false negative</td>
<td>A good part classified as bad.</td>
</tr>
<tr>
<td>false positive</td>
<td>A bad part classified as good.</td>
</tr>
<tr>
<td>feature</td>
<td>An element of an image that is extracted from that image to produce numerical results. For example a line or a circle in the image.</td>
</tr>
<tr>
<td>fuzzification</td>
<td>The process of going from crisp values to fuzzy values.</td>
</tr>
<tr>
<td>fuzzy logic</td>
<td>A method of logical reasoning that is designed to model human linguistic reasoning.</td>
</tr>
<tr>
<td>membership function</td>
<td>A function which determines the degree that individual inputs are part of a given fuzzy set, hence related crisp (non fuzzy inputs to fuzzy inputs).</td>
</tr>
<tr>
<td>multi layer perceptron</td>
<td>A type of neural network that consists of nodes connected to each other so that the inputs travel from left to right in the network.</td>
</tr>
<tr>
<td>neural network</td>
<td>A network of connected neurons that can model a function based on a series of training inputs and corresponding outputs.</td>
</tr>
<tr>
<td>neuro fuzzy</td>
<td>A combination of neural networks and fuzzy logic, typically used in reference to an ANFIS system.</td>
</tr>
<tr>
<td>neuron</td>
<td>The basic unit of a neural network. A neuron accepts inputs and provides an output.</td>
</tr>
<tr>
<td>principle component analysis</td>
<td>The process of remapping data based on creating maximum variance in terms of a given data set.</td>
</tr>
<tr>
<td>receiver operating characteristics graph</td>
<td>A graph used for classify data based on the rate of true positives and false positives.</td>
</tr>
<tr>
<td>QVision</td>
<td>The custom software used to perform visual inspection for this research.</td>
</tr>
<tr>
<td>singular value decomposition</td>
<td>The process of decomposing inputs into principle components.</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In the automotive manufacturing industry the current standard for parts is zero defects. Suppliers cannot ship a defective part. This is enforced by stiff penalties for any company that ships a defective part. Although this is severe it is done for a reason. Especially for safety critical parts, defects are not acceptable. It is no longer a matter of warranties or even the consumer getting what they paid for, it can be an issue of human life. As a result of this need for quality control the demands for zero defects have been implemented throughout the industry to the various tiers of parts suppliers.

The only way to ensure that there are zero defects in a given part is to inspect every part that comes off the line. Typically this is done by manual inspection. This is costly and often ineffective. For a typical worker that cost $20 an hour to the company (with benefits) on a line running 3 shifts a day, with 2 workers inspecting each part on 6-7 lines throughout the plant it can cost $1-2 million dollars a year for quality inspection alone, and this is only in one plant. Furthermore there is no guarantee that the workers will actually catch every defective part. This makes the process very costly with out a guarantee of being effective.

Machine vision has the potential to ensure the parts are properly assembled, ensuring safety of the vehicle. Furthermore, because it’s a computerized system it is not susceptible to fatigue that human inspectors face. Combined with the potential cost savings there is a very good case for the implementation of machine vision systems in the automotive industry. However, it is a difficult problem and the purpose of the thesis is to explore and provide solutions to some of the difficulties in implementing these systems. A working system has been installed on an industrial partner’s assembly as a case study for this research.
1.1 Problem Overview

The industrial partner provides cross car beams for vehicle manufacturers. This is the stamped metal beam that sits behind the dashboard in the vehicle. As part of the assembly process metal fasteners (J-clips and push nuts) need to be inserted into the beam. To ensure that the part is not defective the part must be inspected for the presence of these fasteners. Figure 1.1 illustrates this cross car beam.

![Figure 1.1: Original cross car beam with J-clips and push nuts inserted.](image)

The traditional approach to verify the presence of these fasteners is by checking each beam manually. The human inspector will visually confirm that the fastener is present and then mark it with a marker to indicate presence. This is a very time consuming, hence costly, process. It doesn’t guarantee 100% success. Human inspectors can make mistakes as the industrial partner has found in the past. By implementing a vision system manual inspection can be avoided. Figure 1.2 shows a worker manually inspecting the parts with the manufacturing cell located behind the worker. The worker visually confirms the presence of the clips and marks each clip with a pen.
As a result of the issues with manual inspection the industrial partner installed a vision system on a line making cross car beams for the Ford F150 trucks (46 fasteners). This system was a commercial system that the company installed in house. The system took over a year to setup and still didn’t perform well enough to convince the company to replace human inspection. As a result a project was initiated to investigate this problem. A previous Masters student investigated this problem (Killing 2007). The goal was to produce an inspection system that could be tuned quickly and provide excellent results. Killing developed QVision, a software system for intelligent inspection. After Killing finished his thesis the industrial partner began to commission a new assembly line to produce cross car beams for the Econoline Van. At their request, a vision system based on QVision was installed to verify the presence of 4 J-clips on the radio bracket of the cross car beam. Work on this new application served as the basis for this new thesis.
1.2 Thesis Objectives

The goal of this work is to further understanding in regard to machine vision systems and intelligent inspection techniques. To achieve this goal the work needs to be proven for an actual industry inspection task.

As a case study, a working machine vision system was installed on one of the industrial partner’s assembly lines. The system was required to verify the presence of 4 J-clips on the radio bracket of a cross car beam. The goals in terms of the performance of the system and the demands of the system are based upon the standards set by the industrial partner. This provides an industry based problem as motivation for this research. There must be zero False Positives (FPs), which are defective parts that have been shipped. There must also be less than 2% False Negatives (FNs), which are good parts flagged as bad (a measure of the scrap rate). It is hoped that with the automated system these results can be achieved.

As part of this objective two very different approaches for vision-based inspection have been examined. The first is a feature-based technique, where features such as lines, holes and colour based features (to name a few) are extracted from an image. The image is then evaluated based on the parameters of these features. The second approach considers the entire image as a matrix and processes it as an Eigenimage using linear algebra techniques to redefine the data as a series of coefficients of component images that make up the image. The image can then be evaluated based on this data. These techniques need to be evaluated to determine which is better.

Finally the goal of this work is to apply this knowledge in terms of a software system that can be used as tool for intelligent inspection. Ideally it will be flexible and not limited solely to a single application. It should also be quick to train and configure.

1.3 Organization of Thesis

This thesis is organised into six chapters. Chapter 1 provides a brief introduction to the problem at hand. Chapter 6 consists of conclusions and recommendations.
Chapter 2 provides background on lighting and image acquisition, feature extraction and a variety of classification methods. Recent work in the field of automated visual inspection (AVI) and related areas is also explored.

The results of Chapter 3 are an expansion on the paper “Effect of Lighting on the Performance of a Machine Vision System” (Miles, Surgenor and Killing, 2008). In this chapter the problem of lighting in machine visions systems is explored through the use of a laboratory test cell. QVision, a software package for the inspection of images, is also introduced. Additionally the results of the test cell images, as classified using QVision, are presented.

The results of Chapter 4 are an expansion on the paper “Industrial Experience with a Machine Vision System for the Detection of Missing Clips” (Miles, Surgenor 2009). In this chapter the design and installation of a vision system at an industrial partner’s manufacturing facility is detailed. Challenges with the installation are presented. Changes to the, ANFIS based, QVision software are also detailed including the addition of new features and new feature selection methods. Live images from the test cell are examined and the performance of the system is evaluated.

In Chapter 5 additional methods are investigated in attempts to make the system not only more robust, but also quicker to train, and in two cases eliminate the need for detecting features from the image. The three new classification methods that have been introduced are: a pure Neural Network, Principal Component Analysis and using a direct Eigenimage approach. These methods have been compared in terms of classification accuracy. Next three new feature detectors have been applied. These are a Hough rectangle extractor, a PCA colour based processing method and a corner detector. These are also compared to the previous features used in terms of accuracy of classification. An additional data set has been used to test the performance of these the classifiers and new features in detecting orientation defects in addition to presence and absence of clips.

Appendices containing supporting materials are included at the end of thesis.
Chapter 2

Background and Literature Review

This chapter will provide background on lighting and image acquisition, feature extraction and a variety of classification methods. Recent work in the field of automated visual inspection and related areas is also explored.

2.1 Machine Vision for Inspection

Machine vision (MV) systems for automated visual inspection (AVI) are becoming more commonly used in industry (Davies 2005). A wide variety of industrial applications can be found in the literature. For example, Lee, Li and Daley (2007) apply a vision system to bird handling for the food industry. Reynolds, Campana and Shetty (2004) look at solder paste inspection for the electronics industry. Gayubo et al (2006) apply an AVI system to locate tearing defects in sheet metal.


Although the range of applications is broad, they all tended to adopt the same image processing system with four main stages. The first is image acquisition. This is followed by preprocessing of the image, including applying various filters and selecting regions of interest. The third stage is feature extraction where individual features are extracted from the image. Finally a classifier is used to determine whether a given part is acceptable or not.
2.2 Lighting the Object

Image acquisition is the first stage in a machine vision system. It is essential to obtain the best possible image at this step as all the other stages of the inspection rely on the acquired image. This stage is characterized by camera selection and positioning, lens choice, and lighting design.

The camera should be chosen based on the resolution required for the system and paired with an appropriate lens to capture the details of interest on the object. Factors to consider when selecting a camera include cost, reliability and the operating conditions.

It is very important to choose the lighting for the system properly. Miles, Surgenor and Killing (2008) demonstrate the effect of changes in lighting on the performance of the system. It is shown that when the lighting is changed the performance of the system degrades. This can be compensated for by using automatic shutter, gain and colour correction with the camera. It is also possible to extract features invariant to lighting conditions. Ohba (2000) explores this approach, which is discussed later.

Pham and Alcock (2003) talk about the necessity of proper lighting for AVI. The three main options are front lighting, structured lighting and backlighting. They can be diffuse or directional lighting. Figure 2.1 illustrates these types of lighting and their various subclasses.

Front lighting is when the light source is in front of the object to be lit. Directional front lighting can be broken down into two subclasses of brightfield and darkfield. Brightfield illumination has the camera at an angle to the object being inspected. Brightfield illumination is used to detect surface features. Darkfield illumination has the camera directly facing the object with the lighting source having a low angle of incidence towards the object being inspected. Darkfield illumination can detect small surface features such as scratches and finishes.

Backlighting is when the light source is placed behind the object being inspected. This produces an outline of the object. This is useful for checking the properties of the edges of the object. An example of this is the threads on a bolt.
Structured lighting is used to see 3D aspects of the object. A pattern of lines or stripes is projected on to the object. The 3D height of an object can be determined by how much these lines or stripes are deflected.

Reynolds, Campana and Shetty (2004) discuss these various techniques. They use a structured lighting technique that employs a laser line generator to project lines onto an object. The purpose of this system was to illuminate areas of interest for the inspection of solder paste for electronics manufacturing. They were able to measure the height of the solder paste using this setup.

Ramachandram and Rajeswari (2004) used a structured light source to illuminate objects to be picked up by a robot. The position and pose of the object could be found after a laser pattern was projected onto the object to illuminate a variety of features.

Proper lighting design is essential for proper inspection and classification of object. However it is not an easy task. Lighting design is closer to an art than a science. The approach taken depends on the object being inspected and the details being looked for. It is essential to properly design the lighting system.


### 2.3 Feature Based Methods

The next stage after image acquisition and preprocessing is typically feature extraction. Features such as lines, circles, corners and colours are extracted from images. These features can then be used as inputs into classifiers, which make inspection decisions.

#### 2.3.1 Hough Transform

The Hough Transform is a method that can be used to locate lines in an image (Hough 1959). First a region is selected from a colour image. This region is then converted into a grey scale image. See Figure 2.2 a) and b). After this the image is transformed into the Hough space. The equation of a line can be reparameterized as follows:

\[
\rho = x \cos(\theta) + y \sin(\theta) = \rho
\]  

(2.1)

In this case \( \theta \) is the angle normal to the slope of the line. The other parameter is \( \rho \), which is equal to the y intercept of the line times \( \sin(\theta) \). For every pixel at coordinates \( x \) and \( y \) in an image there are corresponding values of \( \rho \) and \( \theta \). To transform the image into Hough Transform space, every single pixel \( (x, y) \) votes on all the lines (parameterized by \( \rho \) and \( \theta \)) that it can be part of. Each point in the Hough space represents a given line. The point that receives the most votes is the most prominent line. Four images in Figure 2.2 illustrate this transformation process. Figure 2.2 a) shows the original image of the clip and the bracket. Figure 2.2 b) shows the greyscale representation of this image. Figure 2.2 c) shows an image of the gradient Hough transform. This is an image of accumulator, which is the array that contains all the votes cast for the various lines in the image. The brightest peak has been circled in red. Figure 2.2 d) shows the line that has been found by the Hough transform superimposed on an image of the clip. The user is referred to Hough (1959) for more details on this transform. The Hough transform is detailed in both Davies (2005) and Gonzalez and Woods (2008).

The Hough transform is commonly computed after edge detection has been performed on an image.
Figure 2.2: The Hough Transform process; a) region of interest from input image, b) image converted to greyscale, c) image of the gradient Hough transform of the grey scale image. The location of the line (the brightest peak) is circled in red and d) the line found by the Hough transform highlighted on the image of the clip.

Davies (2005) also details the application of this transform to the detection of circles and ellipses. The equation is transformed in terms of radius of the circle and the x and y location of it’s centre or for an ellipse the two axis a and b along with the x and y location of the centre of the ellipse.

An extensive survey of the Hough transforms was conducted by Leavers (1992). The Hough transform as described above is considered. In addition to this a class of probabilistic Hough transforms is also described. For both classes of the transform, performance considerations (processing time and memory requirements) are reviewed. Both hardware and software solutions are explored for this feature detector.

Various methods have been explored to speed up the process of applying the Hough Transform. Duquenoy and Taleb-Ahmed (2006) explore a way to determine the results of the Hough transform by only examining a portion of the image, finding the most prominent line while only partially processing the image.

It is possible to calculate the Hough transform based on a directional gradient, only allowing pixels to vote for lines that have the same directional gradient as the given pixel. Illingworth and Kittler (1987) use this technique to find circles. A vote was cast for a given centre of a circle if the directional gradient of a pixel went through that centre. Peng et al (2006) applied this method for finding the circular outline of nano-scaled glass beads.
Also covered in Davies (2005) and Leavers (1992) is the Generalised Hough Transform (GHT). The GHT is a more complicated algorithm than a standard Hough transform, but also has the ability to find arbitrary shapes.

One of the extensions of the GHT is used to find rectangles. The symmetry in a rectangle can be used to help determine its location. First the gradient direction and magnitude of all the pixels in the greyscale image are calculated. Say the rectangle has side lengths A and B, where $A > B$. The sides A are oriented in the direction of the major axis of the rectangle. The sides B are oriented in the direction of the minor axis of the rectangle. Assume that major axis is oriented between 0 and 90°. By having votes cast for a line B/2 pixels away for pixels oriented parallel to the major axis and A/2 pixels away for pixels oriented parallel to the minor axis it is possible to find the centre of the rectangle.

This process needs to be repeated with a new accumulator under the assumption that the major axis is oriented between 90° and 180°. The highest peak in either of the two accumulators is the location of the centre of the rectangle with the given major axis orientation. This is detailed in Davies (2005).

Jung and Schramm (2004) used the Hough transform to find rectangles in images. First lines were detected in Hough space and then rectangles were found from these lines. There needed to be two sets of similar sized parallel lines. The two sets of lines needed to be at right angles to each other. If these criteria are satisfied it was concluded that a rectangle was found. Davies (2005) notes that there may be greater noise because a two-step process is used. The GHT can increase the accuracy of the shapes found, by using a single step, as opposed to finding individual lines first and then searching for polygons based on those lines to construct polygons.

2.3.2 Corner Detectors

Corners are another set of features that can be extracted from images. Davies (2005) notes that it can be desirable to use corner detectors in the case of non-symmetric polygons, or concave
polygons. He also notes that in such case it may be simpler to find the corners in the object and then apply a pattern matching routine, instead of using the GHT directly.

One method of finding corners is to use a second-order-derivative-based corner detector, which calculates the horizontal curvature multiplied by the intensity gradients to locate corners (Davies 2005).

It is also possible to use a median detector for finding corners. A median filter is applied to an image and then the filtered image is subtracted from the original image. The resulting image contains the corners. A skimming algorithm can be applied where the image is thresholded by the edge gradient. This method has better experimental success than the second order derivative based methods (Davies 2005).

Noble (1988) examines the Plessey corner detector. This corner detector is based on finding $<I_x>$ and $<I_y>$, which are the first derivatives in an n x n neighbourhood. From these values $<I_x^2>$, $<I_y^2>$ and $<I_xI_y>$ can be determined. The Eigenvalues of the matrix

$$
\begin{bmatrix}
<I_x^2> & <I_xI_y> \\
<I_yI_x> & <I_y^2>
\end{bmatrix}
$$

are $\mu_1$ and $\mu_2$. The Hessian of these values ($\mu_1,\mu_2$) is a reasonable measure of corner strength. This detector has been applied to both real and fabricated images with success. It also has the ability to find double corners (i.e. the intersection of lines on a checkerboard). Davies (2005) notes that this detector has difficulty locating the centre of a ‘T’ junction when there is no symmetry in at least one direction of the corner.

A final method that is mentioned by Davies (2005) is the use of the Hough transform for corner detection. This method has the ability to detect corners that may not be entirely square and is based on finding an L shape with the GHT. This has been applied to the detection of the corners of biscuits.
2.3.3 Colour-Based Features

Gonzalez and Woods (2008) detail many methods for image segmentation based on colour techniques. Colour images can be divided into Red, Green and Blue (RGB) components. Each pixel has a red, green and blue component with the combination of the components determining the colour of a pixel. RGB is one example of a colour space or colour model. This is not the only colour space that can be used however.

HSV is another commonly used colour space. The three components are Hue, Saturation and Value. Hue is a measure of the ‘colour’ used whereas the Saturation and Value represent the levels of black and white in the colour, which is similar to the shade of the colour. Figure 2.3 shows an HSV circle. In this circle Hue is represented by a degree value (varying around the circle) and Saturation (the amount of white), which goes from very white at the centre of the circle to less white at the edge of the circle. Value would be measured in the z direction going into the page. As the value increases the colour gets blacker and darker.

Figure 2.3: HSV circle with Hue and Saturation components shown.
Segmentation is the process of separating objects of interest from the rest of the image. One form of segmentation is colour based segmentation. Areas of similar colour can be grouped together. This can be done in the RGB colour space or the HSV colour space or other colour spaces. In the RGB colour space pixels with similar Red, Green and Blue values can be grouped together. For the HSV colour space, pixels with similar Hue, Saturation and Value levels can be grouped together. These colour objects can be quantified based on the number of pixels and the variation in the colour of the pixels.

Lee, Li and Daley (2007) used a Principal Component Analysis (PCA) based approach to transform the original colours into a new basis that provides greater definition of certain colour based features. Applying this technique made it possible to differentiate between features with similar colours. This increased the segmentation performance.

Ohba (2000) experimentally verified that the hue level in the HSV colour space is invariant to lighting provided that colours are far enough from grey. The hue value combined with an Eigenwindow method was used to find objects in a scene invariant to illumination.

### 2.3.4 Textures

Textures can also be extracted from images. Gonzalez and Woods (2008) talk about techniques for this. Kumar (2003) uses texture information to examine defects in fabric. 25 pixels extracted from a 7x7 window around the pixel of interest were used to characterize the texture of a given pixel for this purpose.
2.3.5 SIFT

Scale Invariant Feature Transform (SIFT) is a method for object detection that is claimed to be invariant to translations, rotations, and scale changes between images. Lowe (1999) details this approach. Keys are found in images to represent prominent features. They are identified by using a Gaussian kernel for a Difference of Gaussian (DoG) edge detection method. An image pyramid is constructed with each layer having an effective smoothing of $\sigma = 2$. Maxima and minima of the DoG values are determined from an image pyramid to find these key points. There are approximately 1000 key points for a 512x512 image. The magnitude of the gradient at these key points can be found. The direction can also be found. The values are found by using bins in a local histogram. These keys can be matched to keys in other images to identify objects invariant to scale, position and rotation. This technique has been shown to have success in identifying even partially occluded objects.

2.3.6 Template Matching

Template Matching is the process of looking for a high correlation between a template and an image. A template is tried at all the possible locations in the image and the correlation with the template at each point in the image is calculated. If rotation and translation are looked at all the positions of the template and all the possible rotations of the template also need to be examined. The point with the highest correlation will be where there was the best match to the template. More details on this approach can be found in Gonzalez and Woods (2008). Davies (2005) points out some of the difficulty with this approach. Most significantly there is a high computational overhead associated with processing all the possible positions. This is especially true for a large template.
2.4 Principal Component Analysis

The following is a brief overview of Principal Component Analysis and the reader is referred to Jolliffe (1986) for greater detail on the subject. Principal component analysis is a technique used to reorganise a set of input data into classes that are ordered from maximum variation to minimum variation. By remapping a matrix of inputs based on the eigenvectors of the input's values correlation matrix it is possible to generate new classes of data. Depending on how it is applied PCA can be viewed as a processing of sets of extracted features. Alternatively in the case of the Eigenimage technique it can be viewed as a special type of feature extractor because it can produce numerical outputs directly from the pixel data of an image.

Figure 2.4 illustrates the idea of PCA. Figure 2.4 left image shows the original input data. Figure 2.4 right image shows an illustration of the remapped output data after PCA has been applied. By remapping the data, the variance on the \( x_2 \) axis has been significantly increased and the variance on the \( x_1 \) axis has been decreased. If this data was being used for classification purposes it may now be possible to make the entire classification based upon only the \( x_2 \) values of the point. For this two dimensional dataset it amounts to rotation and translation of the data. In a larger dimensional space this would be a more complex linear transform.

Figure 2.4: left image = Dataset before applying PCA, right image = after applying PCA.
The PCA technique can be very useful for reducing the size of an input dataset. PCA can be applied to rank the variables in terms of variance. The first few components (those with the highest variance) will likely account for a large portion of the variance in the data. By only using these components processing the dataset becomes a lot easier because many of the principal components that don't significantly contribute to classifying the data are eliminated.

**Calculating the PCA’s**

Consider a vector of random variables \( \mathbf{x} \). If you know the covariance matrix \( \Sigma \) of \( \mathbf{x} \) it is possible to calculate the PCA’s. They are defined by \( \mathbf{z} = \mathbf{A} \mathbf{x} \) where \( \mathbf{A} \) is the matrix of eigenvectors of \( \Sigma \), such that:

\[
\lambda \mathbf{A} = \lambda \Sigma \quad (2.3)
\]

where \( \lambda \) are the Eigenvalues of \( \Sigma \). The principal components are ordered by decreasing size of their associated Eigenvalues. The singular value decomposition technique provides a means of calculating the principle components.

**Singular Value Decomposition**

The Singular Value Decomposition (SVD) uses the sample covariance matrix \( \mathbf{S} \) (calculated from the supplied data) and finds the principal components based on \( \mathbf{S} \)

\[
\mathbf{S} = \overline{XX}^T \quad (2.4)
\]

where, \( \overline{X} \) is \( X \) calculated around the sample means or \( \overline{X} = (X - \mathbf{c}) \) where \( \mathbf{c} \) is the mean of \( \mathbf{X} \). Once \( \mathbf{S} \) is calculated the Eigenvalues and eigenvectors of \( \mathbf{S} \) can be calculated and the principal components can be found based on these. The reader is referred to Joliffe (1986) Section 3.5 for further details on this.
Application to Image Processing

One of the areas where employing the PCA technique is useful is for reducing the number of input features. Given an input matrix of features, by applying PCA a new smaller set of features with higher variance can be found. Using these features a classifier such as a Neural Network or an ANFIS network (both discussed later) can be trained.

Eigenimage Approach

Another tool that PCA provides in terms of image processing is the ability to generate “Eigenimages” for object classification. Grey scale images can be represented as vectors of pixels. In this way the dataset $X$ can be generated by vectors of images where $X = [x_1, x_2, x_3 \ldots x_n]$ for $n$ sample images. SVD can then be performed identifying the principal components $[e_1, e_2, e_3, \ldots e_n]$.

These Eigenimages or principal components are a series of images that when combined are able to represent the entire dataset of images. It is often desirable to only select the principal components that have the largest variances $E = [e_1, e_2, \ldots e_k]$. In this case it is possible to project a given image onto the Eigenspace constructed from these principal components. This projection generates an Eigenpoint from the image $x$ with a set of project coefficients $p = [p_1, p_2, \ldots p_n]$. This projection is calculated by:

$$p = E(x-c)$$  \hspace{1cm} (2.5)

Once this projection is known the set of points $p$ can be used to classify the images. Note that $p$ presents a smaller set of input data than the entire image, which presents benefits computationally. In this way the PCA technique acts like a feature detector, producing numerical values from an image.

This theory of EigenImages is discussed further in Sun, Sun and Surgenor (2007) and Ohba and Ikeuchi (1997).
2.5 Processing Methods

Typically in a machine vision system it can be desirable to use a soft computing technique for evaluating information collected from images. Neural Networks and Fuzzy Logic can be employed to provide a system that will deal robustly with highly complicated problems. Neural Networks are trainable networks operating similarly to the neurons in the brain. By using Fuzzy Logic it is possible to express the input data in linguistic terms such as a very large or a small input and evaluate data in those terms. The Adaptive Neuro Fuzzy Inferencing System (ANFIS) is a technique that combines the trainability of Neural Networks with the flexibility in expressing inputs that fuzzy logic gives. By selecting the input features properly the performance of the system can improve. In addition PCA can be used to process input data either for immediate evaluation or input into one of the above systems.

2.5.1 Neural Networks

Neural Networks (NN) are trainable systems that can learn from sets of example inputs and outputs, to form a system that will produce outputs from various inputs. They are similar to the Neurons in the brain that adapt and change their connections based on input stimuli. It is a soft computing technique that can be used to make decisions.

In terms of computer vision applications, features extracted from an image can be used as inputs to train the network based on example pass or fail images. Features can then be extracted from new images. These features can be fed into a Neural Network to classify these new images as pass or fail.

A Neural Network consists of nodes and connections. Figure 2.5 shows an example NN. Each of the nodes takes an input and produces an output. These outputs are then sent through connections to other nodes in the network to eventually produce a final output for given inputs. Each connection is weighted so any input traveling along it is scaled by the given weight. By adjusting these weights it is possible to train the network to give certain outputs. The nodes themselves will only fire if they receive a large enough input. The input to a given is node is the sum of the all the inputs from all the connections to the node after the weights have been applied. The node
is often modeled by a sigmoid (s shaped) function. This function will produce a small output (close to 0) for small inputs (less than 0.5) and large outputs (close to 1) for large inputs (more than 0.5). The following equation shows an example of a sigmoid function.

\[ f(x) = \frac{1}{1 + e^{-2x}} \]  

(2.6)

A step function could also be used, but the sigmoid function has the benefit of being continuous.

Figure 2.5: Sample Neural Network with two layers.

Figure 2.5 shows a standard feed forward network multilayer perceptron network (MLP) with the inputs starting at the left side of the network and moving to the right. The network is a two layer network (a layer being a group of nodes). On the left are the inputs, the layer in the middle is a hidden layer and the right most layer is the output layer. There are different ways of counting the layers in a network, but for the purposes of this thesis, the network in Figure 2.5 will be described as a two layer network because the left most nodes don’t do any processing.

This network can be trained by updating the weights to produce desired outputs from given inputs. One of the techniques used to train a network is called backpropagation. It works by inputting a sample into the network. If the desired output is achieved no updates are made, however if the output is incorrect the weights of the network are updated starting with the weights
to immediate left of the output layer and moving backwards (or to the left) layer by layer to
update the weights immediately to the left of the given layer. As more and more inputs are used
to train the network the ability for the network to correctly classify new inputs should increase.

There are many different network designs including MLP networks, Hopfield networks, radial
basis network and fully connected networks. There are also different ways to train these network.
The MLP network using back propulsion to train the weights is the most commonly used
network. The field of NNs is an extensive topic and the reader is referred to Mehrotra, Mohan
and Ranka (2000) for a more thorough background on Neural Networks.

2.5.2 Fuzzy Logic

Fuzzy logic is a method of logical reasoning that is designed to model human linguistic
reasoning. Instead of a temperature being 20°C it can be labelled using linguistic terms such as:
hot, cold, very hot or not very hot. This allows a system to function in terms of these labels
producing a powerful method of modeling. Fuzzy Inferencing Systems (FIS) are able to process
inputs based on fuzzy logic, making them a valuable tool for use in intelligent systems. Below is
a summary of fuzzy logic. The reader is also referred to Jang, Sun, and Mizutani (1997) for
further background on this subject.

Membership Functions

Fuzzy logic is based on fuzzy sets. A crisp set would contains a series of number such as {-3, - 1,
2 , 4, 7} and any given number is either in the set of not. Fuzzy sets are defined by membership
functions (MFs). These provide a membership degree to each element in the set. For example:
{0.2/-3, 0.4/-1, 0.9/2, 1/4, 0.3/7}. In this case 7 has a weighting or membership degree of 0.3.
Two of the most common Membership Functions (MFs) are the bell shaped function and the
Gaussian function. Figure 2.6 shows these functions. Note how the MFs are defined by
parameters so that they are adjustable.
Fuzzy Operators

As in regular logic, simple operators such as \textit{and}, \textit{or} and \textit{not} can be defined for fuzzy logic. These operators are defined relative to the MFs or fuzzy sets that they operate on. By using these operators fuzzy \textit{if-then} rules can be defined.

Fuzzy if-then rules

In order to generate a FIS it must be possible to evaluate rules based on fuzzy logic. These rules are typical Fuzzy \textit{if-then} statements. An example of this is if the apple is ripe then it is also tasty or \(A \rightarrow B\).

Fuzzy Reasoning: Single Input, Single Rule

It is often useful to make statements about a result using approximate reasoning. For example if the apple is ripe than it is tasty could be a fuzzy rule. So if the apple is ripe you know it is tasty. However if the apple is more or less ripe, then what can be said about it being tasty? Linguistically we can reason that if the apple is more or less ripe than the apple is more or less tasty. This can be extended to fuzzy logic by taking \(A'\) as an input to the statement \(A \rightarrow B\), where \(A'\) is close to \(A\).
Input: \( x \text{ is } A' \)

Rule: \( x \text{ is } A \text{ then } y \text{ is } B \)

Results: \( y \text{ is } B' \)

For this set of reasoning a minimum rule is used to determine the rule firing strength or weighting. So the firing strength, \( w \), or the degree of commonality between \( A \) and \( A' \) is determined by the maximum of the intersection of these two fuzzy sets. \( B' \) is determined by the firing strength. \( B \) is capped at the firing strength \( w \) and the resulting output is \( B' \). Figure 2.7 depicts this process graphically.

![Diagram of Fuzzy Reasoning](image)

Figure 2.7: Diagram of Fuzzy Reasoning for a Single Input Single Rule System with the rule if \( x \text{ is } A \text{ then } y \text{ is } B \) on the left and if \( x \text{ is } A' \text{ then } y \text{ is } B' \) on the right.

**Fuzzy Inferencing Systems**

Fuzzy reasoning can also be extended to a case with multiple inputs and also with multiple rules. This allows for a complex system of fuzzy if-then rules to be implemented for a variety of inputs. For an intelligent system it’s desirable to take input values and produce output values. Typically these are crisp (non-fuzzy) numeric inputs and crisp numerical outputs. Fuzzy logic can be used by transforming input values in MFs (fuzzification), applying *fuzzy reasoning* to these MFs and then finally employing a defuzzification technique to output crisp values. In terms of a vision system this typically means taking numerical values from extracted features, applying fuzzy rules and then outputting a classification between 0 and 1 to determine whether the image is closer to 1, which is a definite pass or 0, which is a definite fail.
Generating MFs

There are many techniques for generating MFs. Ideally an expert user, when setting up the system, can pick MFs if there is some prior knowledge of the inputs. An example of this a product that is typically classified in varying sizes with parameters for small, medium and large. MFs can be generated from this data. Sometimes MFs are not always known and the input data needs to be classified. There are a variety of input partitioning methods available. Three of these are grid partitioning, tree partitioning and scatter partitioning. Once the data is partitioned MFs can be generated from this data.

Rules also need to be generated for a FIS system. These rules should be designed by an expert user according to the system requirements. For example in a fuzzy speed controller a rule could be: if the speed is fast then the input to the motor is low. In the case where there is not enough information about the system to generate appropriate rules it may be advisable to use an Adaptive Neuro-Fuzzy Inferencing System (ANFIS), which is discussed later. It can be trained with input data, so the user does not need to identify MFs or rules.

Sugeno FIS

One type of FIS is the Sugeno FIS. It employs rules that are of the form:

$$\text{if } x \text{ is } A \text{ and } y \text{ is } B \text{ then } z = f(x,y),$$

where \( f(x,y) \) is a crisp function that takes \( x \) and \( y \) as inputs. A simple example of one of these rules is If \( X \) is small then \( Y = 0.15X + 0.02 \). For a two input system \( f(x,y) \) would be a two input function.

To generate the crisp output, the output of all \( f(x,y) \)s are weighted according to the input firing strength. The outputs of all the rules are summed to produce a final output value. Since this output value is crisp there is no need to employ a defuzzification algorithm for this FIS system. Figure 2.8 shows the network structure for a typical Sugeno FIS structure.
The Adaptive Neuro Fuzzy Inferencing System (ANFIS) was developed by Jang (1993). It has the ability to be trained and updated the same way a NN is, but it is a fully fuzzy system. A summary of ANFIS is provided below. However, the reader is referred to Jang (1993) and Jang, Sun and Mizutani (1997) for greater background into this technique.

The ANFIS architecture consists of a five layer fuzzy network structure that is trained by a hybrid learning rule. It is equivalent to a FIS with the additional ability to be trained. Figure 2.9 demonstrates a typical Sugeno-type FIS (with layers 2 and 3 combined). The input data is fuzzified, rules are applied, these rules are normalised, the outputs functions $f(x,y)$ are processed, producing crisp outputs and the final values are summed to produce an output. Details on this structure are presented below.
Layer 1

This is the input layer. Inputs are given to the network. A computer vision example of this could be features extracted from an image. In this layer the inputs are divided up into input MF’s. This process is known as fuzzification. These MF’s are typically bell shaped with a minimum of 0 and a maximum of 1. For example:

\[
\mu_{A_i}(x) = \frac{1}{1 + \left( \frac{x - c_i}{a_i} \right)^2 b_i} \tag{2.7}
\]

where \{a_i, b_i, c_i\} are the parameter set for the MF. A Gaussian MF could also be used. These parameters are adjustable and reflect the linguistic label given to \(A_i\).
Layer 2

This layer is the rule layer. The outputs of the nodes in this layer are calculated by multiplying the inputs of the nodes together for example:

\[ w_i = \mu_{A_i}(x) \times \mu_{B_i}(x) \]  \hspace{1cm} (2.8)

The outputs \( w_i \) are the firing strength.

Layer 3

This layer is the normalization layer. The firing strengths are normalized in this layer so the output is for example:

\[ \bar{w}_i = \frac{w_i}{\sum w_i} \]  \hspace{1cm} (2.9)

Sometimes this layer is combined with Layer 2 and hence only a four layer network is shown.

Layer 4

This is the defuzzification layer. The fuzzy inputs from the rule layer are turned into crisp inputs. For a Sugeno-type system this is a centroid based defuzzification algorithm. Where

\[ O_i^4 = \bar{w}_i f_i = w_i \left( p_i x + q_i y + r_i \right) \]  \hspace{1cm} (2.10)

Note \( \bar{w}_i \) is the normalized output from the third layer. \( \{p_i, q_i, r_i\} \) are tuneable parameters for this layer.
Layer 5

This layer is the summation layer, which is simply a sum of all the inputs for layer 4. This will typically be an output between 0 and 1 for the given network.

Training the Network

The classic way of training a network structure is by using a gradient descent method. For a multilayer network this means taking two passes, applying back propagation on the backwards pass. As in the case of a NN, first an input is propagated forward through the network producing an output value. This actual output value is compared to the target value and an error $E$ is produced. This error is then passed backwards through the network. Moving from the right to the left the error is divided amongst the nodes in the layer to the left of the output layer based on weights of the individual layers. These errors are then divided amongst the nodes of the layer to the left of the current layer. This process repeats until the left most layer is reached. The weights or the parameters of all the nodes are updated to reflect these errors.

With the gradient descent method the errors are updated after each input until the system stabilizes. There are two main drawbacks to this method. The first is the speed of the computation and training and the second is the tendency for the network to get stuck in local minimums.

To deal with this Jang (1993) proposed a hybrid learning method. For the 5 layer network described only the input MF’s and the Rules can be updated. An input is propagated forward through the network as in the traditional gradient descent backpropagation method. This error is then propagated backwards through the network. The rules from layer 4 are then updated via a least squares fitting method and the input MF’s are updated via gradient descent.
Practical Implementations

MATLAB® offers ANFIS routines for the creation and training of an ANFIS network through the Fuzzy Toolbox. These routines use a Sugeno fuzzy system as described previously. The training method used is the hybrid training method previously described.

One thing that is needed to start the ANFIS training is an initial FIS on which to base the ANFIS system. This provides the initial set of input MFs. There are many ways of doing this and three of the typical approaches are grid partitioning, subtractive clustering, and fuzzy C-means clustering. These three algorithms are implemented in the MATLAB® Fuzzy Logic Tool Box as genfis1, genfis2 and genfis3 respectively.

2.5.4 Feature selection

When using a feature based classification scheme it is often necessary to reduce the number of features input to the classifier. It would be computationally expensive and likely not feasible to train a Neural Network on 20,000 feature inputs. In such systems it is also very common to overfit the data when training. This results in poor performance when using the system to classify other data (Hua, Tembe and Dougherty 2009). It is also possible to improve the system performance by reducing the feature set (Killing, Surgenor and Mechefske 2008).

There are a variety of methods of reducing the feature sets that fall into two categories. The first are methods that are classifier independent which use a statistical property of the input data to select or remove given feature inputs. Hua, Tembe and Dougherty (2009) detail a list of 8 of these. The second set of methods are classifier dependent. This involves training the classifier using various groups of features and selecting the best set of features based on the performance of the classifier. These methods include the sequential forward selection (SFS), the branch and bound method, and the sequential forward floating selection method.
SFS works, by choosing the best single feature based on classifier performance. This feature is held and then the classifier evaluates all the combinations of this feature with one other feature. These two best features are then held. All the combinations of these two features are examined with an additional feature. The number of features is increased in this manner until the desired performance is reached or all the features are used.

Pudil, Novovičová, and Kittler (1994) propose a slightly modified SFS method. They point out that one of the difficulties with the forward search is once a feature is selected it cannot be unselected. This means the results will likely be suboptimal. They propose the following sequential forward floating selection method. If n features are currently held for step 1 the set is increased using the SFS search described above. For step 2 the least significant feature is found in this new data set. If this is the feature that was just added, then keep the feature and continue from step 1 again. Otherwise remove the feature. For step 3 find the next least significant feature. If removing this improves the performance repeat step 3. Otherwise don’t remove the feature and go back to step 1. This method has the advantage that features can be removed after they are selected.

Garcia, Villalobos and Runger (2006) also explore the problem of feature selection. Multivariate stepwise discriminate methods are used to choose the appropriate features for classification. This is a statistical filter approach for selecting the appropriate subset of features to classify the data.

### 2.6 Receiver Operating Characteristics Graphs

Receiver Operator Characteristic (ROC) graphs can be generated to compare false positives and false negatives at different classification thresholds (Fawcett, 2006). The vertical axis is the True Positive Rate (the ratio of correctly classified pass clips to total pass clips). The horizontal axis is the False Positive rate (the ratio of false positives to the total number of failed clips). A true positive rate of 1 is desirable and a false positive rate of 0 is desirable. The ROC graph plots all the True Positive Rates vs False Positive rates for every threshold possible. The threshold ($t_{\text{threshold}}$) is the value at which any result higher than the threshold is classified as a pass and any result lower than the threshold is classified as a failure.
Ideal results for a ROC graph are results that touch the top left corner. What this indicates is that there is a threshold at which it’s possible to have a True Positive rate of 1 and a False Positive rate of zero. For these ideal results the line goes from the bottom of the graph to the top of the graph vertically showing that the True Positive rates increases without the False Positive rate increasing i.e. 0 FPs (note the threshold is dropping from 1 to 0 in this case). Then the line turns and goes horizontally across the top of the graph because as the threshold drops eventually every clip is classified as a pass. Most importantly if the graph touches the top left corner there is a threshold where there will be no FPs and no FNs.

A diagonal line is presented on the ROC graphs. This represents a coin flip. If the ROC graph ever goes below this diagonal line the results are very poor. In that case the ratio of True Positives to False Positives would be less than 0.5. At that threshold the results would be better if a coin was flipped to determine whether each case was a pass or fail.

Figure 2.10 below shows two sample ROC graphs for clip classification. Figure 2.10 left image shows poor classification results whereas Figure 2.10 right image shows excellent results. The closer the data is to the top left corner the better the performance of the graph. The reader is referred to Fawcett (2006) for more details ROC analysis.

![ROC Graphs](image)

Figure 2.10: Sample ROC graphs left image = poor classification results, right image = excellent classification results.
2.7 Previous Work

2.7.1 Clip Detection

Three graduate students have looked extensively at the problem of J Clip detection on cross car beams. They are Killing (Queen’s University), Sun (University of Calgary) and Mehran (Concordia University). Below is a detailed summary of the previous work done on this problem by these students.

Killing Summary

To tackle the problem of clip inspection Killing et al (2006) identified an initial training set of images. These were from a stamped cross car beam, from an industrial supplier. They consisted of both push nut images and J clip images. These were to be inspected by a machine vision system to determine their presence.

A mask was generated to locate the push nut, by using ANFIS methods and Mahalanobis approaches for colour detection. The ANFIS method showed better performance, correctly classifying all the clips, whereas the Mahalanobis only correctly classified 96.4% of the pass clips and 100% of the fail clips out of a set of 600 pass images and 100 fail images.

Killing, Surgenor and Mechefske (2007) extracted features from the machine vision images. Lines, holes and circles were found in the image along with colour information. The lines were extracted by using a Hough transform based on the gradient of the image. Circles were also extracted using a similar method. The holes were found by using a standard Hough transform, after edges were extracted using a Sobel edge detection algorithm. The colour areas were found by identifying the number of pixels of a given hue within a predefined region.
These features were turned into inputs by taking the angle of the line, number of pixels in a line, radius of a circle etc. This data was then compiled into a training matrix. Using this matrix as an input it was possible to train an Adaptive Neuro-Fuzzy Inferencing System (ANFIS). Once the ANFIS system was trained it was possible to evaluate live images based on this system. Features were extracted from these live images. Inputs were then created from these features i.e. the angle of lines, the number of pixels on a line, the radius of a circle. These features were then input into the ANFIS system and the image was evaluated as either a pass or a fail.

To further improve the performance of the ANFIS system, the input feature set was reduced. By using only certain features it was possible to improve the performance of the system. To test this a laboratory test cell was constructed. Using all the features available there were 4 false positives (FPs), which are fail images classified as pass. There were 0 false negatives (FNs), which are pass images classified as fail. Finally there was an RMS error of 0.094. When only certain features were selected there was an RMS error of 0.0056 and all the clips were classified correctly. This was for a set of 100 pass and 100 fail images.

Killing, Surgenor and Mechefske (2009) investigated this same problem. With 49 live images from the manufacturing plant including abnormal faults the optimized ANFIS system had zero FPs and 3 FNs. For an ANFIS with no optimization there was 1 FP and 0 FNs. Using an unoptimized threshold classifier there were 2 FPs and 1 FN. With an optimized threshold classifier there was 2 FPs and 4 FNs. FPs are unacceptable to the manufacturer, because a defective part is shipped whereas FNs slow production (it’s a measure of the scrap rate). For this test the ANFIS solution had 0 false positives and hence the best performance.

**Sun Summary**

A similar database of clip images was inspected by Sun, Sun and Surgenor (2007). Their approach to verifying the presence of these clips involved a PCA based technique. A Region of Matching (ROM) was located. The user defined this as a rectangle region where the clips could be found in all images. Using this ROM a Region of Verification (ROV) was found inside the ROM. This was done by using template matching for edge pixels. The Geometric Model Finding routine in Matrox Imaging Library was used to locate the ROV from an edge image.
After the ROV was located a PCA based approach was applied to the ROV region. The pass images were used to construct a set of principal component Eigenimages. Only the most prominent Eigenimages (the ones with the highest corresponding Eigenvalues) were used. Images could be approximately reconstructed based on their coefficients in Eigenspace.

New images were also decomposed into Eigenimages. The error between the original image and reconstructed image was calculated by taking a sum of the residual squares between the two images. By analyzing this error it was possible to classify an image as either a pass or a fail by defining an appropriate threshold for this error. The original Eigenimages were constructed from pass images, so fail images had large reconstruction errors.

Online training was used to train the model. New images were added to the test set and manually labelled. This was continued until the mean and standard deviation of the reconstruction error stabilized.

This PCA model was applied to an image set. The 8 largest EigenImages were used which consisted of 80% of the eigenvalues. The system was able to correctly classify all the rejected clips when online training was performed and misclassified clips were incorporated into the system for a set of 1595 pass images and 4 fail images.

In further work by Sun, Sun and Surgenor (2008), this same method was applied to the inspection of J Clips. The system was trained online. In total 32 false negative cases were found that were incorporated into the PCA model. After 1200 parts had been inspected the system had stabilized and no more false negatives were found.

**Mehran Summary**

A fuzzy set theory approach has been used to look at a collection of J clips by Mehran, Demirli, and Surgenor (2006). The data is similar to those used by Killing and Sun. There were 706 images of which 647 were pass (clip present) and 59 were fail images (missing clips). A subset of these images was created using 55 pass and 55 fail images. Initially, to try and detect defects, a variety of global statistical approaches were applied to the images in question. The methods used were: Mean comparison, Variance Comparison, Euclidian Distance Comparison, Covariance
Comparison, along with Histogram Comparison using Absolute Error and Least Square Error. For the mean comparison there were 3 images that could not be classified as either a definite pass or a definite fail, for variance there was 1 image, for Euclidian Distance Comparison there were 4 images, for Covariance Comparison there were 7 images, for Histogram Comparison there were 15 and for Least Square Error there were 31. Although they did provide some classification information, none of these methods were able to completely separate the data set into pass and fail images.

An XOR model was defined to examine the images. A composite pass image was created and a composite fail image was created for the Red, Green and Blue components of the image. These composite images were the average pixel values of all the pass or fail images respectively. An XOR operation was then applied on both an input image and the composite image. Images with a low result were considered to be similar to the given image. White and Black XOR operators were defined comparing given images to a white and black background. By using White, Black, and Present and Missing XOR Operators for the red and green composite images it was possible to generate a fuzzy system. By using a Sugeno system it was possible to correctly identify all of the 706 images. It was noted that ANFIS did not lead to a better result than the Sugeno model, although no numerical results were given for ANFIS.

2.7.2 Fuzzy Logic

Lee and Bien (1997) have looked at finding objects in images using fuzzy logic. They present a fuzzy corner detector as a feature extractor. These corner features are then grouped into line segments, which are then described using fuzzy membership functions. Finally these segments are input into a FIS system to calculate the transformation of the object based on a library model. They also suggest automatic tuning of the FIS system should be investigated. The object in question is an artificial image created for these tests.

Lashkia (2000) used X-rays to detect defects in welds. These X-ray images were classified using fuzzy reasoning. Features were extracted from the image and fuzzy rules were generated for determining whether a weld is improperly classified. A Neural Network is employed for tuning the fuzzy system. Note that this is not same as the ANFIS approach as detailed by Jang (1993).
Mehran, Demirli and Surgenor (2006) also employed fuzzy logic to classify XOR functions for the purpose of clip detection.

2.7.3 Neural Networks

Kumar (2003) has applied a Neural Network system coupled with PCA for detecting local textile defects. Features were extracted from the image and then processed with PCA. The components from the PCA are used to train a feed forward Neural Network. The results found for this NN were satisfactory for defect detection and for web inspection. These results were evaluated on real fabric samples.

Hua, Tembe and Dougherty (2009) have examined the use of a Neural Network classifier to evaluate feature selection techniques. The input data set was a microarray chip that contained 20,000 or more features. Filter based feature selection was conducted and then classifier specific feature selection was conducted using the NN as a classifier.

Ramachandram and Rajeswari (2004) used a NN to help position a five degree of freedom robot manipulator. Features were extracted from an image of an object and then these were used as inputs into a NN. An MLP feedforward network was used with backpropagation to train the network.

Kwak, Ventura and Tofang-Szai (2000) used a Neural Network approach for defect detection in leather. Defects were detected by image segmentation including morphological processing. These defects were used as inputs into a NN. This NN was then used to classify the features into five different types (lines, holes, knots, stains, and wear defects) based on input features. An MLP network was chosen, with backpropagation for training the network. It was found that a 2 layer network (1 hidden layer) had better performance, ranging from 96.25% to 95% accuracy, for this task than a 3 layer network (with 2 hidden layers), which ranged from 96.25% to 92.5% accuracy.


2.7.4 ANFIS

As noted above Killing (2007) has presented a feature-based system that uses ANFIS to process the features from images. Mehran, Demirli and Surgenor (2006) have also investigated ANFIS for processing XOR operators on an image, but did not find significant improvements.

Polat and Güneş (2007) used a PCA based ANFIS approach for determining diseases based on lymphography data. A dataset based on the lymph system was used as inputs to classify lymph related diseases. First PCA was applied to the dataset. Fuzzy preprocessing was then used to weight the principal components. These values were subsequently used as inputs into an ANFIS network. They produced an 88.83% accuracy of classification compared with accuracies of classification ranging from 81.08% to 77.03% for 4 other methods.

Avci and Turkoglu (2009) proposed a PCA based ANFIS approach as well. This was applied to analysing signals from Doppler techniques generate for determining whether a heart value disorder was present. First features were extracted from the signal. PCA was then applied to the input features. The principal components were then supplied as inputs to the ANFIS system. The classification accuracy for this system was 96% for normal subjects and 93.1% for abnormal subjects.

2.7.5 PCA

Kumar (2003) presented a Neural Network based approach for the detection of textile defects. Features were extracted from the images. PCA is performed on these features as a preprocessing step before evaluating them based on a Neural Network.

Lee, Li and Daley (2007) presented a method of colour analysis by performing PCA on the individual colour channels of an image. The colours were reclassified and then colour based
segmentation was performed on these new colour components. This has been effectively applied to inspection of food products.

As already reported Sun, Sun and Surgenor (2007) used an Eigenimage technique for the inspection of clips on a cross car beam. Ohba and Ilkeuchi (1997) proposed an Eigenwindow method breaking an image into small windows where PCA is performed to detect objects. This method was able to detect partially occluded objects. Ohba, Sato, and Ikeuchi (2000) presented work where this same Eigenwindow technique is applied on the Hue component of an HSV image for illumination invariance.

Gayubo et al (2006) also used a PCA Eigenimage based technique to find defects in sheet metal. The inspection camera was mounted on a robot to allow the camera the ability to move. First PCA was used to find the optimal positioning of the inspection robot before taking the final image for inspect. By comparing the original image of the part to a set of library images taken from various robot positions the optimal position can be found. After the optimal position is found the final image for inspection is acquired. This image is then mapped to the Eigenimage Space and classification is made based on the coefficients in the Eigenimage Space.

2.8 Summary

There are a variety of methods available for the inspection of objects using AVI systems. These can be divided into two main categories. The first is the class of feature based inspection approaches. The second is based on principal component Eigenimages.

A variety of classifiers have been proposed including Fuzzy based, ANFIS and MLP NNs. ANFIS and Fuzzy based classifiers are very similar in that they both produce a FIS. The advantage of ANFIS is that it can be trained.

For the feature based approaches features are extracted from an image producing numerical inputs for a classifier. As noted two commonly used classifiers are ANFIS and MLP NNs. These can be used directly or with feature selection. One of the modifications of this approach is to apply PCA first and then use the principal components to train the classifier.
For the EigenImage approach images are mapped into Eigenspace, providing a set of coefficients for every image in Eigenspace. These coefficients can be used for classification. They can then be classified in a similar way to the feature based inputs by using either ANFIS or an MLP NN for the classifier.

Based in part on the review of the literature what is proposed is a comparison of three main methods with two classifiers. The three methods are 1) feature based, 2) feature based with PCA applied, and 3) the Eigenimage approach. The classifiers are 1) ANFIS and 2) MLP NN. The methods will be compared in order to determine which performs best for classification when applied to the inspection of a radio bracket as an industry motivated case study.
Chapter 3

Laboratory Test Cell Results

The results in this chapter are an expansion on the paper “Effect of Lighting on the Performance of a Machine Vision System” (Miles, Surgenor and Killing, 2008). In this chapter the problem of lighting in machine vision systems is explored through the use of a laboratory test cell. QVision, a software package for the inspection of images, is also introduced. The results of these test cell images, as classified using QVision, are presented.

3.1 The Problem

Currently the standard for product quality in the automotive industry is zero False Positives (FPs), which is a bad part labeled as good and hence incorrectly shipped to the customer. Suppliers are not allowed to ship a defective part. As a result of this companies are investing significant resources into defect detection. One area companies have looked to solve these problems is Automated Visual Inspection (AVI). AVI systems have been around for decades, but there are still many difficulties in their practical applications. Designing the correct lighting and properly tuning the system are essential, but difficult tasks.

An industrial supplier of stamped metal parts installed a machine vision system on one of their product lines. The purpose of this system was to detect the presence of metal fasteners that are attached to the stamped parts. The idea was to eliminate the need for costly and inconsistent manual inspection. Not only does the manual inspection cost up to $60 000 per month for this line alone, but humans make mistakes. Factors like fatigue can cause their performance to change over the course of a shift and from day to day. The company hoped that by installing this system they would be able to accurately inspect these parts.

Unfortunately this system took over a year to setup and even then did not have satisfactory results. In the end it was turned off and replaced with manual inspection. Figure 3.1 shows manual inspection of a cross car beam. To deal with these issues a research project was initiated to develop a system that is flexible, accurate and easily tunable.
3.1.1 The Bracket

The specific area of inspection is the radio bracket. The purpose of this part is for securing the radio in the vehicle. This is a stamped metal part that is part of the cross car beam forming the metal frame behind the instrument panel. In order to be able to attach the radio, metal J clips are clipped onto the stamped metal part so that the screws can fasten the radio in place. There are four J-clips on the radio bracket and inspection must be able to detect their presence or absence. Figure 3.2 shows an image of the radio bracket.
3.1.2 The Approach

To systematically approach this problem the research has been completed in stages. The first stage consisted of research into the inspection problem in a laboratory setting. A test cell was created for investigating the inspection of the radio bracket. Various lighting systems were installed in this cell to test the effects of lighting. A custom software package *QVision*, written with MATLAB®, has been used to inspect these parts.
3.2 QVision software (Killing)

As part of his Master’s Thesis Jonathan Killing developed the QVision software package see Killing (2007). A graphical user interface (GUI) was constructed to assist in the classification of parts. Using this GUI an expert user could select features, train the system and further improve those results by having the system automatically select the most relevant features. The details of this algorithm along with screen captures of the program are presented below as a point of comparison for the extensive improvements and developments on this software package since that time.

The QVision software package allows an expert user to identify features such as lines, holes and colours in training images. These features can then be used to train an ANFIS system. This FIS structure can then be “optimized” where the results are improved by only choosing certain input features to improve the performance of the classification.

The software consists of three modules. The first is the Configuration Module for feature classification. The second is the Training Module for training the ANFIS system. The final module is the Camera Module. This module is used for evaluating live images with the system.

3.2.1 Configuration Module

The configuration module is used to load the library data images and define features for these images. Using a GUI it is possible to identify lines, circles, holes and colour regions in the image. Using these features as inputs it should be possible to train the system and classify clips as either present or absent. Figure 3.3 shows a screen shot of this module.
Figure 3.3: Image of Configuration Module.

Lines

Lines are detected by a using a Hough gradient method. First a colour image is converted into a grey scale image. The gradient direction and magnitude of every pixel in the grey scale image are then calculated. The slope of any line perpendicular to a given pixel is known from the gradient direction. The accumulator can then be filled by each pixel voting on any line it may be part of based on the slope of the line and the position of the pixel. The vote is weighted by the magnitude of the gradient at that point. The line with the most votes is the one selected.

The selected line is classified by three parameters: $\rho$, $\theta$ and $\tau$. $\rho$ is the $y$ intercept of the line multiplied by the sin of $\theta$. $\theta$ is the angle of the line perpendicular to the line found. $\tau$ is the strength of the line, which is the total accumulated votes for the line. $\rho$ and $\theta$ are parameters that determine the position of the line. Strength or $\tau$ is how well the line is defined. A very well defined line will have a high strength value. A poorly defined line will have a low strength.
Circles

Circles are found using a Hough circular gradient method. The image is first converted to grey scale. After this the gradient direction of every pixel is calculated. The accumulator is generated by having each pixel vote for any circle that the pixel could be part of. A pixel is part of a circle if the directional gradient of the pixel points from the pixel to the centre of that circle. If a pixel’s edge direction is pointing to the centre of a given circle, that circle gets a vote. If not then it doesn’t. The weightings of these votes are determined by the magnitude of the gradient at each pixel. Figure 3.4 illustrates this process.

The x and y positions of the centre of the circle, the radius of the circle and the strength of the circle are returned. The strength of the circle is the total weight of all the votes received by that circle.

Figure 3.4: Illustration of circular gradient detection method.
Holes

Holes are found by first converting the image to a grey scale image. After this a Sobel edge
detector is applied to find the edges in the image. A circular Hough transform is then applied.
This transform is based on pixels as opposed to gradients. For every pixel in the image a vote is
cast for any circle, which that pixel could be a part of. The weight of each vote is the intensity of
the pixel. The circle with the most votes is then chosen.

The x and y position of the centre of the circle, the radius of the circle and the strength of the
circle are returned as parameters. The strength of the circle is the total weight of all the votes the
circle received.

Colours

Colours are identified from images based on their hue. Hue is a component of the HSV colour
space and gives value to the colour of a pixel independent of the shade of that pixel. In order to
identify blobs of colour the image is segmented into n hue ranges of equal size (hues being values
from 0 to 360 degrees). For example if there are 6 hue regions, all hues from 0 to 60 are in the
first region, all hues greater than 60 and less than or equal to 120 are in the next etc. To find the
colours any pixels that aren’t in the given hue range are ignored. Noise is then removed from the
remaining pixels by using a morphological opening function followed by a morphological closing
function. The remaining blobs of colour are then sorted based on their respective areas. The hue,
area and centroid of the largest blob of colour are returned. If no hue range is selected the largest
blob of colour in any of the n hue ranges is returned.

Relative Features

It is also possible to find relative features. First a line, circle or hole is found. Then other
features are searched for based on the location of the line, circle of hole. This can be a useful tool
if there are easily found features close to the area being inspected. The area of inspection can be
found relative to distinctive features on a part such as holes or lines.
3.2.2 Training Module

Once the features have been defined in the configuration module they can then be extracted from the image and used as inputs to train an ANFIS system.

Input Data

The features are extracted for every image in the training set. They are labeled as either pass or fail based on whether they are extracted from a pass or fail image. Figure 3.5 shows a screen shot of this. After this the data needs to be divided into training, checking and testing subsets. The default is 40% training, 20% checking and 40% testing. This is done by randomly assigning images to these subsets. All the extracted features from a given image go into the set the image was assigned to. In order to generate an ANFIS system the data then needs to be grouped into membership functions (MFs). This is done using fuzzy subtractive clustering using the MATLAB® genfis2 routine. Figure 3.6 shows sample MFs for one set of input data.

![Image of input data to the system](image)

Figure 3.5: Sample input data to the system. The blue upside down triangle represents the data point for a pass image and the red triangle represent an input for a fail image.
Figure 3.6: Sample membership function. The blue and red triangles represent pass and fail images respectively. The curves represent the membership function. The degree of membership in an MF for any given point is between 0 and 1.

ANFIS

Once the data is divided into membership functions it can then be trained. This is done using the `anfis` routine in MATLAB®. This routine trains the network and will only work with a Sugeno-type FIS structure. QVision automatically uses a Sugeno FIS structure so that the `anfis` routine will work. Figure 3.7 shows a sample diagram of the network structure for an ANFIS system. The training set is used to train the ANFIS network and the checking set is used to ensure that the training is correct.
Figure 3.7: Sample Network for ANFIS system.

Sample Outputs

After the network has been trained the data should be tested using the testing set. When the features values are input into the ANFIS system numerical results, typically between 0 and 1, are produced. The results for the testing set can then be plotted. Figure 3.8 shows sample training results. A correctly classified pass image should be as close to 1 as possible and a correctly classified fail image should be as close to 0 as possible.
Figure 3.8: Output results of training. Green dots represent correctly classified images.

**Feature Selection “Optimization”**

By selecting the input features to the ANFIS system carefully it is possible to improve the results of the classification. There are often inputs that contribute very little to appropriate classification of the system. Often when these are removed the performance of the system improves. QVision uses a Forward Search, based on Root Mean Square (RMS) output error of the checking data, as the basis for whether a given optimization is better than another. Figure 3.9 shows a selection of available optimizations for the system.
3.2.3 Camera Module

The real time module interfaces with the camera (see section 3.3.3 and Appendix A for specifications). It is able to acquire a live image and then classify this image as either pass or fail. To make this classification the features defined in the configuration module are extracted and input into the ANFIS network from the training module. It is possible to do this in real time. Figure 3.10 shows a screen shot of the Camera module.
3.3 Test Cell

A test cell was constructed to test the effects of lighting on the performance of a machine vision system. The goal was to study the problem in a laboratory setting under controlled conditions as preparation for the installation of a vision system on the actual assembly line.

3.3.1 Layout

The cell is pictured below in Figure 3.11. The bracket is sitting in front of a white background. There are three different lighting setups: fluorescent lights, LED lights, and a camera mounted LED ring light. The camera is set back from the part. Figure 3.12 shows a layout of the test cell from the top. The positions of the lighting systems are shown, along with the camera and the bracket. The part is located 0.46m from the lens.
Figure 3.11: Image of test cell.

Figure 3.12: Layout of the test cell, showing the three different lighting solutions.
3.3.2 Lighting

Three different types of lighting solutions were investigated with the test cell. The first type is a fluorescent bulb. There are two fluorescent lamps with a plastic diffuser. The second type is an LED spotlight. There are also two of these. There is no diffuser on these spotlights. The final lighting solution was an LED ring light. This was mounted on the camera and also had a plastic diffuser. Figure 3.13 shows sample images results under the varying lighting conditions. Appendix A contains specifications for all three lighting solutions. Changing the lighting changes the radiance, which is the amount of light that shines on the clip.

![Figure 3.13: Image with 3 different sources of light: left – Fluorescent, middle – LED spotlight, right – LED ring.](image)

3.3.3 The Camera

The camera used for these experiments is a CCD IMI Tech-1080FT colour camera. The resolution to capture the images is 640x480 pixels. Further details on this camera can be found in Appendix A. The camera has the capability of auto-adjusting shutter speed (i.e. time of exposure) and the gain of an image. These features are controlled by feedback loops that measure the difference between the average whiteness of an image versus a reference signal. The camera also has the ability to provide auto white balance for the images. This changes the ratio of red to green and green to blue in the image. These features can be turned off if necessary. The goal of these features is to have the camera adapt for various changes in lighting. Using the auto features amounts to controlling the radiance, which is the amount of light entering the camera.
Using auto settings can be very advantageous, but they should be used with caution. If the auto features are used to capture an image it is uncertain exactly what settings were used to capture a given image. There is also no guarantee that the auto features will produce constant images and may even produce erroneous results. If the system produces wrong results it will be difficult to tell whether this is an image acquisition problem or a processing issue, making troubleshooting difficult. The best way to deal with the problem is to provide a lighting solution that will provide constant irradiance (the light entering the camera) by keeping the radiance (the lighting) constant.

### 3.4 Experimental Work

#### 3.4.1 Data Sets

For testing purposes four different sets of images were obtained. The room lights where either on or off for the trials and the camera auto-adjust settings were varied between trials. The room lights were turned off to change the effect of the lighting on the radio bracket.

As already mentioned in reference to Figure 3.12, three types of lights that were used to illuminate the images are: fluorescent, LED spot and LED ring. The differences in these lighting conditions can be seen in Figure 3.13. There is more blue present in the image when the LED light sources are used. The fluorescent lights provide more yellow in the image. The fluorescent lights are the brightest followed by the spotlights, followed by the ring light.

There were a total of 18 images in each data set: 9 pass images and 9 fail images. There were 3 images taken with fluorescent lighting, 3 images taken with LED spotlights and 3 images taken with LED ring light.

Features were selected on these data sets. Figure 3.3 shows these features defined on a clip. There is a line on the top of the clip, a line to the left of the clip, a circle feature and a colour feature.
Figure 3.14: Ring light on, room lights on, camera auto off (pass image on left, failed on right).

Figure 3.15: Ring light on, room lights off, camera auto off (pass image on left, failed on right).

Figure 3.16: Ring light on, room lights on, camera auto on (pass image on left, failed on right).

Figure 3.17: Ring light on, room lights off, camera auto on (pass image on left, failed on right).
3.4.2 Performance Measures

Two ANFIS networks were trained and tested, with four different trials. The performance of the classifiers was evaluated on the basis of four indices: i) RMS error, ii) number of false positives, iii) number of false negatives and iv) number of uncertain. A false positive occurs when the system passes a part that is actually missing a clip. A false negative occurs when the system fails a part that actually has the clip present. In practice, false negatives are permitted (it’s a measure of the scrap rate). However, false positives are not permitted. Thus, systems tend to be set up to err on the side of false negatives.

For the purposes of this study, the root-mean-squared (RMS) error $E_{\text{rms}}$ for a set of images is used as a performance measure. The root-mean-square of the output error defined as:

$$E_{\text{rms}} = \sqrt{\frac{\sum_{i=1}^{n}(Z_i^d - Z_i)^2}{n}} \quad (3.1)$$

where $Z_i^d$ is the desired (correct) classification for the $i^{th}$ image, $Z_i$ is the output of the classifier algorithm ($Z_i^{\text{ef}}$ or $Z_i^{\text{th}}$) and $n$ is the total number of images. $Z = 1$ is an unconditional pass (clip present) and $Z = 0$ is an unconditional fail (clip missing). For a classifier, output $Z$ values close to 0.5 are considered as uncertain, as they represent a situation where the output values activate two conflicting rules simultaneously. Thus, images are classified as uncertain for the condition:

$$B_L < Z_i < B_U \quad (3.2)$$

where $B_L$ is the lower bound and $B_U$ is the upper bound on the classifier output. In this study, unless stated otherwise, adopted values were $B_L = 0.45$ and $B_U = 0.55$. A false positive is for the condition when $Z_i^d = 0$ and $Z_i > B_U$. A false negative is for the condition when $Z_i^d = 1$ and $Z_i < B_L$. For most applications, it was found that fixed values of $B_L$ and $B_U$ were adequate for determining classifier outcome. However, adjustment of the values could shift the output of the system to more conservative results, or a preference for false positive, or a preference for false
negative outputs. Thus, $B_i$ and $B_f$ should be considered as tunable parameters as the particular values used will depend on the type of data being observed.

### 3.4.3 Results

The results for the four trials are summarized in Table 3.1. In Trial 1, the first network was trained with images taken with the camera auto-adjust settings off and the room lights on. Figure 3.14 illustrates two images from this set. This Network was then tested with images taken with the camera auto-adjust settings off and the room lights off. Figure 3.15 illustrates two images from this set. Compared with the images from Figure 3.14, there is a significant increase in blue. The images in Figure 3.15 are also dimmer and there are more shadows compared to the images from Figure 3.14. Note that the shutter speed had to be lowered with respect to Figure 3.14 to achieve a reasonable exposure. In both cases the shutter speed was adjusted manually to try and provide a similar brightness of the clip and the beam with respect to the images that used auto-settings (Figure 3.16 and Figure 3.17).

**Table 3.1: Summarized performance test results for the four trials.**

<table>
<thead>
<tr>
<th>Trial</th>
<th>Test condition</th>
<th># False Negatives</th>
<th># False Positives</th>
<th># Uncertain</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Camera auto off, UNoptimized</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0.4972</td>
</tr>
<tr>
<td>2</td>
<td>Camera auto off, Optimized</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0283</td>
</tr>
<tr>
<td>3</td>
<td>Camera auto on, UNoptimized</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0.1563</td>
</tr>
<tr>
<td>4</td>
<td>Camera auto on, Optimized</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0004</td>
</tr>
</tbody>
</table>

In Trial 3, the second network was trained with images taken with the camera auto-adjust settings on and the room lights on. Figure 3.16 illustrates two images from this set. The background colour is now white and the exposure is a little darker. This network was then tested with images taken with the camera auto-adjust settings on and the room lights off. Figure 3.16 illustrates two images from this set. The background colour is still white and the images are slightly brighter...
than the images in Figure 3.16. Note that there are similar amounts of shadow in images for Figure 3.16 and Figure 3.17.

### 3.4.4 Feature Selection Results

It is possible to improve the performance of the ANFIS network, by removing features that don’t help in the classification of the image. This was done using a SFS algorithm based on RMS results. This provides a simpler network and in many cases, more accurate results. Trial 1 repeated with an optimized network is given as Trial 2. Trial 3 repeated with an optimized network is given as Trial 4.

Figures 3.18 through 3.21 illustrate the effects of this feature selection. Figure 3.18 is the network for Trial 3 (no feature selection). The inputs on the left correspond to features highlighted by the operator. The output on the right is a value between 0 and 1. A value of 1 is a definite pass and 0 is a definite fail. Outputs should be on (or near) 0 or 1. Outputs near 0.5 mean the system is having difficulty classifying the data. Figure 3.19 gives the classification results for 6 images from Trial 3, using the network shown in Figure 3.18. Figure 3.20 shows the network structure Trial 4 (optimized). Relative to the network structure from Figure 3.18, one sees that the network is much simpler. Features that were determined to have little effect on the classification result were removed by the system. Figure 3.21 gives the classification results for 6 images from Trial 4, using the network structure from of Figure 3.20. The classification results are significantly improved with all points being close to 1 or 0.

For Trial 1 the classification performs poorly with 1 false positive and 4 uncertain results. The RMS error for Trial 1 is 0.497. In Trial 2 all the clips are classified correctly, there are no uncertain results and the RMS error drops to 0.028. For Trial 3 the results are good, but there are 4 uncertain results, and the RMS error has increased to 0.156. Trial 4 shows the best results with all the clips correctly classified and an RMS that has dropped to 0.0004.
Figure 3.18: Network structure for ANFIS system without feature selection.

Figure 3.19: Testing results without feature selection.
Figure 3.20: Network structure for ANFIS system with feature selection.

Figure 3.21: Testing results with feature selection.
3.5 Summary

In conclusion, the system performance is better when the camera auto-adjust settings are turned on. For the networks where feature selection was not used there is a large increase in performance from Trial 1, with the auto-adjust settings off, to Trial 3 with the auto-adjust settings on. The system also performs significantly better when feature selection is used. In both cases by using feature selection uncertainty in classification is reduced and the RMS error is drastically reduced. Thus, the optimal system is with the camera auto-adjust on and the network optimized. The RMS error for the optimal case was practically speaking zero (0.0004). Note that both trial 2 and trial 4 showed very good results. The reason to use the auto-adjust results (trial 4) is because of robustness of the system. The RMS error is lower in trial 4 and the results are better classified. Note also that the shutter settings were manually adjusted for trial 2. Using the camera auto-adjust settings avoids the need to tune this parameter.

A laboratory test cell has been constructed to test a variety of lighting solutions for J-clip detection on the radio bracket of a cross car beam. Fluorescent lights, LED spotlights and LED rings lights have been used as light sources. A CCD camera with optional auto settings has been used to acquire images of the radio bracket.

QVision is a GUI based image inspection software program. By extracting lines, holes, circles and colour features from an image an ANFIS system can be trained. This system can then be used to classify live images. Tests were conducted using the various lighting setups, and the various camera settings. Feature selection was also used to improve the performance of the ANFIS system. When the camera auto settings were used the classification results were more accurate. When feature selection was used the classification results were also more accurate. When both the camera auto settings and feature selection were used the best results were obtained.

To get the best possible results it is recommended that the lighting stay constant. This means having a constant irradiance (the light entering the camera). Firstly, radiance (the light hitting the clip) should be controlled to control the irradiance. Auto settings can be used when needed by compensating for changing radiance. A software solution could also be used to deal with changing irradiance, however keeping the lighting constant should be attempted first. By employing feature selection it is also possible to increase the accuracy of classification results.
Chapter 4

Industrial Manufacturing Cell Results

The results of this chapter are an expansion on the paper “Industrial Experience with a Machine Vision System for the Detection of Missing Clips” (Miles, Surgenor 2009). In this chapter the design and installation of a vision system at an industrial partner’s manufacturing facility is detailed. Challenges with the installation are presented. Changes to the, ANFIS based, QVision software are also detailed including the addition of new features and new feature selection methods. Live images, of production parts, run through the test cell are examined and based on those images the performance of the system is evaluated.

4.1 The Problem

A new line has been commissioned assembling a new cross beam and the industrial partner has requested that a vision system be installed to detect defects in this part. Chapter 3 has looked at the radio bracket in a laboratory test cell environment. This chapter looks at the full cross car beam, which contains the radio bracket, on the running assembly line at the plant.

4.1.1 The Cross Car Beam

The part in question is a cross car beam. It is a stamped metal part that is the supporting metal for the instrument panel and dashboard of a vehicle. This part is shown in Figure 4.1. It should be noted that this is a different part than Killing (2007), has examined. Compare figures 1.1 and 4.1.

As part of the manufacturing process metal push nuts and J-clips are inserted into the beam. These need to be inspected for presence or absence. The clips of particular interest are the 4 J-clips located on the radio bracket, which is a stamped metal part that the radio is to be attached to. The radio bracket is physically part of the cross car beam.
For the production of the new beam the manufacturer has installed a new clipping cell for the beam shown in Figure 4.1. The old cell is still in production, but the original vision system has been decommissioned and has been replaced by a new vision system installed by an outside contractor. It serves as a unique point of comparison to gauge the performance of the work presented here versus the success of the system installed by the outside contractor.

4.1.2 Performance Criteria

The vision system must interface with the Programmable Logic Controller (PLC) controlling the cell and produce accurate results. In order to replace human inspection two criteria must be met. The first is 0 False Positives (FPs). FPs are defective parts that have been passed by the system. Due to the nature of the industry it is unacceptable to ship a defective part. The second criteria is a maximum of 2% False Negatives (FNs). These are good parts that have been failed by the system. Although no harm has been done to the customer, every failed part needs to be re-inspected. This slows down production so a rate of FNs higher than this level is also unacceptable.
4.2 QVision Updates

The original software developed by Killing (2007) for his Master’s thesis has been heavily modified for the inspection of J-clips on the radio bracket. The largest modification has been allowing the software to work with two cameras and four clips. Additional improvements that have been made include, adding features such as the ability to find the length of a line segment, a semi-circle detector, a centroid based algorithm for determining the centre of the bracket, a colour statistics features and a radial hole based statistics feature. A testing module has been included for testing additional datasets on a trained solution. Log files for recording the activity of the system have been created along with a camera setup tool for proper alignment and positioning of the cameras. The system has also been modified to communicate with the Programmable Logic Controller (PLC) that runs the cell via an RS232 serial connection. Additional algorithm changes have been made as well. These are based around feature selection and will be detailed in this chapter.

4.2.1 Four Clips and Two Cameras

One of the first tasks that needed to be carried out was modifying the original QVision so that it could support analysing four clips. To do this it was decided that a single camera would capture an image of two clips. There was a top camera that captured images of the top two clips and there was a bottom camera that captured images of the bottom two clips.

In the configuration module the images were input based on whether they were from the top or bottom image (see Figures 4.2 and 4.3). The feature selection, training and end classification is completed individually for each clip meaning that there are four different ANFIS networks trained. For the configuration module there is a toggle to choose the current clip and features can be selected for that clip. Figure 4.2 illustrates the features defined for Clip 1 (top left clip). Figure 4.3 illustrates the features defined for Clip 3 (bottom left clip).

For the training module there is also a toggle that allows switching between the clips. Figure 4.4 illustrates the modifications to the training module. The camera module was modified to process all 4 clips at the same time. The results for all four clips are displayed individually. This is detailed in Figure 4.5.
Figure 4.2: Configuration screen. The user has the ability to select feature parameters for the four clip positions.

Figure 4.3: Configuration screen showing features being defined on the bottom bracket.
Figure 4.4: Training Module updated to work with four clips.

Figure 4.5: Camera Module configured to work with four clips.
4.2.2 Additional Features

In order to improve the results of classification several new features were added to improve the performance of the system. These are the length of line segments, a semi circle detection method, a centroid based method for finding the centre of the bracket, a colour statistics method and a radial hole statistics method.

**Line Length**

To improve the classifications of lines, the ability to measure the length of a line segment was added. To do this first a line was identified by a gradient based Hough transform. After the most prominent line was found any pixels that belonged to the line were found. These pixels were thresholded by a minimum gradient value of 8, which was the same value used in finding the original line.

After the pixels were identified, they were grouped into line segments based on any pixels that were 8 connected. A pixel is 8 connected with another pixel if the other pixel is one of the 8 pixels surrounding the original pixel. This can be compared to a 4 connected pixel, which is connected if it is directly touching a connecting pixel to the top, bottom, left or right, but not on the diagonals. Figure 4.6 illustrates this, a) shows 4 connected pixels and b) shows 8 connected pixels.

Once these line segments are known they can be classified by the two pixels that bound the segment. The longest line segment is chosen. The length of this segment is the length of the line.
Figure 4.6: a) An illustration of 4 connect pixels. The red and blue objects are not connected. b) An illustration of 8 connected pixels. There is just one connected object.

**Hough Semi Circle**

A Hough semi circle function was added for the purpose of being able to detect semi circles in the image. The routine is similar to the Hough circle detection method. First the direction and magnitude of the gradients of all the pixels in the image are found. These are thresholded so that any pixel with a gradient magnitude less than eight is ignored. The pixels then vote for any circle that they could be part of. If a line normal to the pixel, as determined by the gradient direction, intersects the centre of a potential circle a vote is cast for that circle. Votes are weighted by the magnitude of the gradient of the pixel.

For the semi circle detector a filter stage is added before the votes are cast. Either a top or bottom semi circle must be classified. Only pixels that are part of the given semi circle can cast votes. For a top semi circle provided an edge point is above the potential centre of the circle the vote is counted. The reverse is done for a bottom semi circle. At the end the semi circle with the most votes is selected. Figure 4.7 illustrates the process of detecting a semi circle. To extend this process to an arbitrary portion of a circle would not be difficult. The filtering step would need to be modified to include the given arc of the circle.

An approach such as calculating the ratio of edge points to the circumference of a circle could be considered, but it will not be robust to noise. A circle where every other pixel was not included because of noise would register the same as a semi circle.
Centroid – Finding the Beam

In order to give a reference point for finding other features, a centroid feature has been used to find the centre of the beam. This feature works by finding the centroid of the image. The user selects a region of interest in the image and filtering is applied. This region of interest is selected once and is the same for all images in the dataset. After the filtering is applied the centroid is found based on pixel intensity. This is the centre of weight of the pixels in terms of pixel intensity.

Two types of filtering were used. Figure 4.8 shows the centroid algorithm when the image is simply converted to greyscale. Figure 4.9 shows the centroid algorithm when the image is first converted to greyscale, threshold based on intensity and then the centroid is found. The thresholding is based on 50% of the maximum pixel value. Any pixel less than or equal to 50% is ignored and any pixel greater than 50% is used to find the centroid. It was found that 50% was a good cut off in this case, separating the beam from the background as can be seen in Figure 4.9.

This algorithm is not as precise as a template matching algorithm for finding the bracket. However the algorithm requires significantly less computation and is a more general approach.
This generality means that this can be applied to a variety of inspection problems. Any object that is significantly brighter than the background can have its centroid found this way.

Figure 4.8: Illustration of the centroid algorithm for grey scale image, with the ‘+’ sign denoting the centroid of the image.

Figure 4.9: Illustration of the centroid algorithm. This is for a thresholded black and white image with the black ‘+’ sign denoting the centroid of the image.

**Colour Statistics**

A global colour statistics feature was added. This is a statistical colour feature that is able to find the average colours in a given area of the image. The user selects an input area, and based on this, the average red values, the average blue values and the average green values of the pixels in the area, are returned. Additionally the image is converted into grey scale and the average grey scale intensity is returned. The goal of this feature is to be able to find differences in the colour of the images with clips versus the images without clips.
A radial histogram based method has been added. This is a modification of an original technique presented by Davies (2005). After a circle is initially located it is examined moving outwards from the centre of the circle. A radial profile can be generated for the given circle. Based on this profile it is possible to examine changes in values of intensity from inside the edge of the circle, outside the edge of the circle and on the edge of the circle.

To generate the radial data the image is divided into rings a given distance from the centre of the circle. For a circle centred at \( x_0 \) and \( y_0 \) the value of each ring can be calculated by:

\[
I_{\text{ring}} = \int_{\theta_0}^{\theta_2} \int_{r_0}^{r_2} \text{Intensity}(x, y) \, d\theta \, dr
\]  

(4.1)

where,

\[
r = \sqrt{(x - x_0)^2 + (y - y_0)^2}
\]  

(4.2)

For this particular application, the three regions of interest were: 1) the area inside the circle, 2) the area immediately surrounding the circle and 3) the area outside the circle. Given a radius of the circle \( r \) and ring width \( r_w \) the areas of interest are as follows:

\[
\begin{align*}
    r_{\text{inner}} &\leq r - r_w \\
    r - r_w < r_{\text{mid}} &\leq r + r_w \\
    r + r_w < r_{\text{outer}} &\leq r + 10
\end{align*}
\]  

(4.3)

A final parameter \( \text{change} \) is defined by

\[
\text{change} = r_{\text{outer}} - r_{\text{inner}}
\]  

(4.4)

The ring width was typically chosen between 2 and 5. This is half the thickness, in pixels, of the middle ring. The goal is to have a fairly thin ring, so that it only contains all the pixels that are on the edge of the hole. This allows the pixels outside the hole and inside the hole to be compared without using any of the pixels on the hole. The outer radius of the outer ring was chosen to be 10 pixels + the outer radius of the middle ring. This was to place a sensible maximum on the size.
of the ring. This maximum could be changed as needed for the application. Figure 4.10 illustrates the rings of interest for the Radial Hole statistics feature.

Figure 4.10: An illustration of the radial hole statistics feature.
4.2.3 Testing Module

A testing module has been added to the software. The purpose of this module is to be able to test a dataset that is different from the original training set. Datasets are loaded into the testing module and the output is presented in a graphical format identical to the training module. The data is tested based on the trained network from the training module. This module also works with all four clips. Figure 4.11 illustrates this new module.

![Testing Module](image)

Figure 4.11: The Testing Module.
4.2.4 Display Log

In order to record the activities of the vision system a log was added to the program. Every clip that is classified by the program is recorded in this log as either a pass or a fail. The log files are sorted based on the date and the shift they are for. This allows for easy inspection of the operation of the vision system after the fact. If the number of failed clips is less than the permitted number of FNs then the system is running smoothly. If the number is close then the individual images need to be examined. The log file does not allow for verification of FPs. To verify that there are no FPs the images must be examined manually. The images taken by the system are also saved with a time and date stamp in case they are needed for future use.

To easily view the log file a graphical display has been added. Failed clips are conveniently highlighted. This provides comparable behaviour to a similar commercial vision system in the same plant. Figure 4.12 illustrates the log file. One additional feature is the ability to export the files in Excel format for easy evaluation.

![DisplayLog](image)

Figure 4.12: The log file graphical display.
4.2.5 Camera Setup

In order to be able to properly position the cameras, a camera setup screen was created. By using this tool, the user is able to align the cameras so that the clips are within the red rectangles on the screen. If the clips are in these rectangles the cameras are aligned properly. This system runs in real time with a constant video stream so that the focus of the lenses can also be adjusted. Figure 4.13 shows the Camera Setup screen.

![Figure 4.13: The camera setup screen for properly aligning the cameras and adjusting focus.](image)

4.2.6 Interfacing with the PLC

To fully integrate the vision system with the manufacturing cell the vision PC needed to be integrated with the PLC that runs the cell. This is done with an RS232 (serial) connection. The PLC sends the message ‘Trigger Camera’ that tells the vision system to take a picture and then process it. After this the vision system returns either Pass or Fail based on its inspection. The PLC can then respond appropriately and continues its cycle.
4.3 The Cell and Installation at the Industrial Partner

The goal of the project was to install a working machine vision system at the industrial partner’s manufacturing facility. Details of the physical setup are provided below. These include a layout and description of the new cell, the camera and lighting setup, along with sample images from the system. Figure 4.14 shows the software on the PC running next to the PLC, which controls the clipping cell.

![Image of PLC and PC](image)

Figure 4.14: The PLC running the clipping cell next to PC running the vision system.
4.3.1 Cell Layout

A diagram of the clip insertion and testing cell is shown in Figure 4.15. The cell is on a rotating table, with parts being rotated between four stations. The part is loaded and unloaded at Station 4, located at the bottom of the figure. Station 1, on the left, is for robotic clip insertion. Station 2, on the top, is for mechanical clip inspection. Station 3, on the right, is for ID stamping. The cameras for the machine vision (MV) system are located at Station 3 (labelled as 5 in the Figure 4.15).

Figure 4.15: New clipping cell with the vision system located at 5.
4.3.2 Camera Installation

Two MV cameras were installed to view the bracket. They are IMI Tech 1080 FT cameras. Each camera looks at two clips. The top camera looks at the top clips and the bottom camera looks at the bottom clips. A 12mm lens is used for the camera. This size gives the largest possible image while having both clips in the field of view. The cameras are mounted on custom mounting brackets attached to the cell. The cameras are shown in Figure 4.16 a) and the bracket in Figure 4.16 b). Camera specifications can be found in Appendix A.

![Figure 4.16: a) Camera configuration (each camera images two clips), b) The bracket in the clipping cell.](image)

4.3.3 Lighting Solution

For a lighting solution an LED ring light, with a diffuser was chosen (see Appendix A for specifications). This ring light is pictured on the front of the camera in Figure 4.16 a). This decision was made based on the need for a lighting solution that could illuminate the part, but did not have a large footprint due to space considerations in the cell. It also needed to be easily mountable. Because of this it was not feasible to install an off angle lighting source. The lights were manually aligned to illuminate the bracket and be centred on the beam.
4.3.4 Sample Images

Figure 4.17 shows sample top pass and top fail images taken from the cell and Figure 4.18 shows sample bottom pass and bottom fail images taken from the cell. It should be noted that one of the difficulties with the cell is how the ring light illuminates the clips. Depending on the angle the clips are facing they can catch more or less light changing their brightness. This is illustrated in Figure 4.18: left where the clip on the left is brighter than the clip on the right.

The radio bracket in the image can vary in position from cross car beam to cross car beam. The beam is attached to a fixture, but the fixture moves, as part of the operation of the cell, within a tolerance of +/- 2mm. This means the bracket’s location can change from image to image. The clips also have some movement. This makes it difficult to apply a direct template matching approach.

Figure 4.17: left – Top Pass image, right – Top Fail image, from the manufacturing cell.

Figure 4.18: left – Bottom Pass image, right – Bottom Fail image, from the manufacturing cell.
4.4 New Feature Selection work

It is often possible to improve the results of the training of the system by properly selecting the input features. By removing features that do not distinguish significantly between a pass or a fail clip it is possible to increase the accuracy of the results (Killing, Surgenor and Mechefske, 2008 and Hua, Tembe and Dougherty, 2009). A total of four methods are explored for this work. They are all classifier specific methods. For a classifier specific method the features are chosen by how they effect the classification of the data by using a given classifier. In this case it is the ANFIS network. The alternative method is a filter based method that is independent of the classifier and typically works based on statistical properties of the input set.

False positives and false negatives are used as evaluation criteria for accuracy of the system and not RMS. However, RMS error is used in the feature selection because it is more feasible to work with. It is not an unreasonable measure of performance. Solutions with a low number of false negatives and false positives have low RMS error. Conversely, solutions with large number of false positives and false negatives have large RMS errors. Additionally RMS error provides a numerical difference between solutions with identical numbers of false positives and false negatives. RMS error is easily calculable and is a reasonably accurate measure of classification performance.

One note on the discussion in the next section is that the features referred to here are different from the features extracted from image. The features used for feature selection are the attributes of the extract features. A feature would be a line or a circle, whereas an attribute of that feature would be the length of the line or the radius of the circle. To be consistent with literature feature selection for the purpose of this section involves choosing appropriate feature attributes (radius, x location etc.) from image features (lines, holes etc.) to produce the best output solution. Sometimes this process is also called optimisation because the goal is to produce the best classifier performance possible.
4.4.1 Exhaustive Method

Ideally it would be best to test every combination of input feature attributes. However this is not feasible for this system. For example if there are 52 input attributes, that means that $2^{52}$ sets of attributes would need to be considered. This is computationally impossible. For this reason other methods of feature selection have been considered.

4.4.2 Forward Selection Method

A common approach is a forward based feature selection. First, all the single attributes are tested as inputs to the classifier (in this case the ANFIS network). The attribute with the lowest RMS error is selected. This attribute is then paired with all the other possible attributes. These pairs are then input into the classifier. The best combination of the two attributes, in terms of lowest RMS error, is selected. This process then continues matching the set of two attributes with all the other attributes to find the best set of three attributes. This continues until a given error threshold is reached or there are no more attributes to include. It is also possible to specify other stopping criteria if the results stop improving.

4.4.3 Forward (many) Method

One of the difficulties with the forward search is that it is not optimal. The best attribute for classification when used by itself may not be a part of the best solution. Also a given set of attributes may not be a part of the best solution, but be the best the forward search method returns.

It is desirable to increase the scope of the search area that the forward search looks in. To do this instead of keeping only the single best set of attributes at each step the best n attributes were kept. For example say n is 3. All the attributes would be used as single inputs into the classifier and the best 3 attributes would be kept. These 3 attributes would then individually be paired with all the other attributes. Out of all the constructed pairs of two attributes the best 3 would be kept. These 3 sets of two attributes would be paired with all the rest of the single attributes to find the 3 best
combinations of 3 attributes. This process would continue until a certain minimum error is reached or there are no more features to use.

By expanding the forward search in this manner the number of combinations examined is significantly increased, with the search still being completed in a reasonable amount of time.

4.4.4 Exhaustive with Prefiltering Method

The final method examined was based on prefiltering for an exhaustive search. In this method all the attributes were input individually into the classifier as single inputs. The best N of these were selected based on RMS output of the classifier. These N attributes are then searched exhaustively for the best combination of attributes.

In this method it is possible to substitute the classifier based method for a filter based (classifier independent) method to find the best N features. One example of one of these filters is the t-test (Hua, Tembe and Dougherty 2009). However the classifier based methods are expected to be more informative because they depend on how the system is trained. This is in comparison to a filter based method, which uses no information about how the system is trained.

4.4.5 Typical Workflow

With the four methods of feature selection the user has many options for attempting to find a good set of features. There are four choices for the user to try and there are adjustable parameters for these choices. In order to provide some insight to how these tools might be applied a typical workflow has been generated. This is by no means the only way to use these tools, but it is one possible option.

The four available optimization methods and their tuneable parameters are shown in Figure 4.19. Both the exhaustive search and the forward search have no tuneable parameters. For the forward (many) search it is possible to choose how many features attributes are kept at each stage. For the exhaustive search with prefiltering it is possible to choose how many features attributes remain after prefiltering.
A possible workflow that is proposed is shown in Figure 4.20. The criteria for a suitable solution have been left unspecified in this case, but typically it would be 0 false positives and less than 2% false negatives on the given testing set. There is a possibility that there will be no acceptable solutions when following this process and that means either the features need to be redefined or the algorithm itself needs to be improved upon.

In the proposed workflow if less than 10 feature attributes are used it is feasible to perform an exhaustive search. If none of these exhaustive results are suitable then a more significant rework is needed. If 10 or more attributes are used it is suggested that the forward search be run. If this fails, then the forward (many) search should be used. If that search fails the number of feature attributes held at each stage should be increased. If it’s failed three times, while increasing the number of features held each time, then the user should move on to the exhaustive with prefiltering selection method. If that fails then the number of attributes not filtered out should be increased. If the number of feature attributes not filtered out is larger than 12, then the exhaustive
search will not be feasible because of processing time and hence no solutions have been found. It is important to note that this is just one suggested workflow and is an example of what the user could choose to do. QVision permits different workflows.

Figure 4.20: One typical workflow for completing feature selection.
4.5 Results

The system was installed with the heavily modified QVision as detailed in the previous section. The system was commissioned and then used for real time inspection of clips on production cross car beams.

The system showed good performance on the tested data in most cases. However one of the difficulties encountered was a rusty bracket. Figures 4.21 and 4.22 show typical pass and fail clips respectively. Figure 4.23 shows an image with rust present on the bracket. In this case the bracket and clips had rusted and the clip had been manually removed causing a defect. The rust left an outline that looked similar to a properly installed clip on the bracket.

Figure 4.21: Two pass clips, for top bracket.

Figure 4.22: Two failed clips, for top bracket.

Figure 4.23: Two failed clips with rust present, top bracket.
4.5.1 Datasets

A total of four data sets were used for testing purposes. For the first set, the training set consisted of 150 pass images and 114 fail images for the top bracket, and 147 pass images and 125 fail images for the bottom bracket. In addition, 3 testing sets were collected. The second set consisted of 2017 top images and 2016 bottom images. The third set consisted of 45 top fail images and 60 bottom fail images. The fourth set consisted of 24 top fail and 24 bottom fail rusty images.

4.5.2 Results

Table 4.1 contains details of the feature selection results for all four clips. The data presented is based on results with 2017 pass images, 45 top fail images and 60 bottom fail images. The first feature selection (Feature Selection 1) uses all 52 feature attributes as inputs (hence there is no feature selection in this case). The second feature selection (Feature Selection 2) uses only 4 feature attributes as inputs. Note there are no rusty clips in this data set. See Appendix B for a complete description of the data sets and the features used in both Feature Selection 1 and 2.

<table>
<thead>
<tr>
<th>Clip</th>
<th>Feature Selection 1 False Negatives</th>
<th>Feature Selection 1 False Positives</th>
<th>Feature Selection 2 False Negatives</th>
<th>Feature Selection 2 False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td>346</td>
<td>24</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>Clip 2</td>
<td>334</td>
<td>7</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Clip 3</td>
<td>796</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clip 4</td>
<td>226</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The number of false positives for feature selection 2 using only 4 feature attributes is 0, with less than a total of 2% false negatives. For feature selection 1 the results are not acceptable because of the presence of FPs and high levels of FNs.

The results of this classification were presented as RMS scatter plots. The y-axis represents the classification with any value greater than 0.6 being classified as a pass and any value less than or equal to 0.6 being classified as a fail. The x-axis represents which image number the results came from.
from. Ideally the results will be close to either 1 or 0 depending on whether the clip is a pass or fail.

Receiver Operating Characteristics (ROC) graphs were also generated to compare false positives and false negatives at different classification thresholds (Fawcett, 2006). The vertical axis is the True Positive Rate (the ratio of correctly classified pass clips to total pass clips). The horizontal axis is the False Positive rate (the ratio of false positives to the total number of failed clips). A true positive rate of 1 is desirable and a false positive rate of 0 is desirable. The ROC graph plots all the True Positive Rates vs False Positive rates for every threshold possible. The threshold ($t_{thld}$) is the value at which any result higher than the threshold is classified as a pass and any result lower than the threshold is classified as a failure.

Ideal results for a ROC graph are results that touch the top left corner. What this indicates is that there is a threshold at which it's possible to have a True Positive rate of 1 and a False Positive rate of zero. For these ideal results the line goes from the bottom of the graph to the top of the graph vertically showing that the True Positive rate increases without the False Positive rate increasing i.e. 0 FPs (note the threshold is dropping from 1 to 0 in this case). Then the line turns and goes horizontally across the top of the graph because as the threshold drops eventually every clip is classified as a pass. Most importantly if the graph touches the top left corner there is a threshold where there will be no FPs and no FNs.

A diagonal line is presented on the ROC graphs. This represents a coin flip. If the ROC graph ever goes below this diagonal line the results are very poor. In that case the ratio of True Positives to False Positives would be less than 0.5. At that threshold the results would be better if a coin was flipped to determine whether each case was a pass or fail.

Figure 4.24 shows the RMS results for the results of feature selection 1. There are FPs and FNs present for this graph. Figure 4.25 shows the corresponding ROC graph. The line never touches the top left hand corner and hence perfect results are not possible. Figure 4.26 shows good RMS results for feature selection 2. They are not perfect because there are values that are not 0 or 1, however at the given threshold of 0.6 all the clips are classified correctly. The ROC graph touches the top left corner indicating this.
Figure 4.24: Clip 3 No Rust results with no feature selection, presented in a scatter plot.

Figure 4.25: Clip 3 No Rust results with no feature selection, presented in a ROC graph.
Figure 4.26: Clip 3 *No Rust* results with feature selection, presented in a scatter plot.

Figure 4.27: Clip 3 *No Rust* results with feature selection, presented in a ROC graph.
4.5.3 Rusty Clip Results

One of the issues encountered when field-testing the system was the presence of rusty clips. Table 4.2 shows some results with the effects of the rust on clips. The first test No Rust has been optimized for no rusty clips being present. These results are very good with zero false positives and a low percentage of false negatives. When the 24 rusty clip images are added to the top and bottom fail image set, a large number of false positives are found. These false positives were closely examined and as a result a radial hole algorithm was suggested.

With the use of the radial hole algorithm the results have improved, but do not meet the stringent criteria of zero false positives. Table 4.2 also shows results for data sets without rust. There are still false positives present and the number of false negatives has increased significantly, however the solution is closer to being correct. The bottom two clips (Clip 3 and Clip 4) have the best results (no false positives, but the false negatives are still high on Clip 3).

Table 4.2 Results with and without rusty clips.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Clip</th>
<th>FNs</th>
<th>FPs</th>
<th>%FN</th>
<th>Pass Clips</th>
<th>Fail Clips</th>
<th>Total Clips</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Rusty Images</td>
<td>Clip 1</td>
<td>8</td>
<td>0</td>
<td>0.4%</td>
<td>2017</td>
<td>45</td>
<td>2063</td>
</tr>
<tr>
<td></td>
<td>Clip 2</td>
<td>3</td>
<td>0</td>
<td>0.2%</td>
<td>2017</td>
<td>45</td>
<td>2063</td>
</tr>
<tr>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>0</td>
<td>0%</td>
<td>2016</td>
<td>60</td>
<td>2026</td>
</tr>
<tr>
<td></td>
<td>Clip 4</td>
<td>3</td>
<td>0</td>
<td>0.2%</td>
<td>2016</td>
<td>60</td>
<td>2026</td>
</tr>
<tr>
<td>Rusty Images Added</td>
<td>Clip 1</td>
<td>8</td>
<td>11</td>
<td>0.4%</td>
<td>2017</td>
<td>69</td>
<td>2086</td>
</tr>
<tr>
<td></td>
<td>Clip 2</td>
<td>3</td>
<td>24</td>
<td>0.2%</td>
<td>2017</td>
<td>69</td>
<td>2086</td>
</tr>
<tr>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>14</td>
<td>0%</td>
<td>2016</td>
<td>84</td>
<td>2100</td>
</tr>
<tr>
<td></td>
<td>Clip 4</td>
<td>3</td>
<td>18</td>
<td>0.2%</td>
<td>2016</td>
<td>84</td>
<td>2100</td>
</tr>
<tr>
<td>Rusty Images + Radial Feature</td>
<td>Clip 1</td>
<td>494</td>
<td>1</td>
<td>24%</td>
<td>2017</td>
<td>69</td>
<td>2086</td>
</tr>
<tr>
<td></td>
<td>Clip 2</td>
<td>77</td>
<td>19</td>
<td>4%</td>
<td>2017</td>
<td>69</td>
<td>2086</td>
</tr>
<tr>
<td></td>
<td>Clip 3</td>
<td>86</td>
<td>0</td>
<td>4%</td>
<td>2016</td>
<td>84</td>
<td>2100</td>
</tr>
<tr>
<td></td>
<td>Clip 4</td>
<td>19</td>
<td>0</td>
<td>0.9%</td>
<td>2016</td>
<td>84</td>
<td>2100</td>
</tr>
</tbody>
</table>
Figure 4.28 illustrates the features used for Clip 2 in Table 4.3. This provides a visual reference for these features relative to the clip. All the new features introduced in this chapter have been used (including semi-circle and line length), but they were not all kept after performing feature selection.

Table 4.3 Features used in *Rust + New Algorithm*

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features</th>
<th>Clip</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td><em>Line Left – Score</em></td>
<td>Clip 3</td>
<td><em>Line Top – T</em></td>
</tr>
<tr>
<td></td>
<td><em>Hole – Score</em></td>
<td></td>
<td><em>Line Bottom – T</em></td>
</tr>
<tr>
<td></td>
<td><em>Colour Statistics – Red</em></td>
<td></td>
<td><em>Line Diagonal Left – R</em></td>
</tr>
<tr>
<td></td>
<td><em>Colour Statistics – Blue</em></td>
<td></td>
<td><em>Line Diagonal Right – Score</em></td>
</tr>
<tr>
<td></td>
<td><em>Hole Statistics – Out Avg</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>Hole Statistics – Change</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clip 2</td>
<td><em>Line Left – Score</em></td>
<td>Clip 4</td>
<td><em>Line Left – Score</em></td>
</tr>
<tr>
<td></td>
<td><em>Line Right – T</em></td>
<td></td>
<td><em>Line Right – Score</em></td>
</tr>
<tr>
<td></td>
<td><em>Hole – Score</em></td>
<td></td>
<td><em>Line Bottom – Score</em></td>
</tr>
<tr>
<td></td>
<td><em>Line Diagonal Left – Score</em></td>
<td></td>
<td><em>Line Diagonal – Length</em></td>
</tr>
<tr>
<td></td>
<td><em>Colour Statistics – Intensity</em></td>
<td></td>
<td><em>Colour Statistics – Intensity</em></td>
</tr>
<tr>
<td></td>
<td><em>Centroid – Y</em></td>
<td></td>
<td><em>Centroid – X</em></td>
</tr>
</tbody>
</table>

Figure 4.28: Illustration of features used for clip 2 in *Rust + New Algorithm*.  

---

92
4.5.4 Signing off on the System

Typically with a system of this sort the industrial partner will demand that a set of criteria be met for them to ‘sign off’ or accept the system. The system must maintain the specified criteria of 0 FPs and less than 2% FNs for the duration of these tests. The procedure that was developed for this can be found in the Figure 4.29. It is a flow chart detailing this process.

The first step for signing off on the system was to successfully classify 5000 parts with less than 2% false negatives and 0 FPs (note rusty clips were excluded from this criteria by the industrial partner). If there was a failure, changes needed to be made to the system and the process started again. After 5000 parts were successfully classified a Gage R&R test (Repeatability and Reproducibility see Mottonen et al 2008) would be performed. The specifications for this would involve testing the system on a variety of the cross car beams, loaded by multiple operators. Provided these beams were correctly classified the system would be accepted for use.

In the case that there were either false negatives or false positives the images that caused the erroneous results were examined and a new solution was proposed. In the case of less than 5 FPs there was typically no pattern. With so many input variables it’s not always known why certain images failed. After the dataset was reworked the testing would begin again.

The 5000 parts were never successfully tested. The economic downturn in the fall of 2008 caused the automotive industry to suffer a decline in production. Due to this decline it was not possible to finish testing 5000 parts with the latest solution.

There was also difficulty with the other commercial vision system installed in the adjacent cell. The industrial partner has put a significant investment in this system, but that vision system has also failed to replace manual inspection on that cell.

Ideally it would be possible to test 5000 parts and complete the Gauge R&R successfully. This would give the industrial partner confidence in the system and hopefully machine vision in general. This will provide a cost effective solution to the inspection problem. The library images provide an opportunity for continued investigation of this problem.
Figure 4.29: Flow chart detailing acceptance process.
4.6 Summary

It is possible to generate acceptable results on the given data set without rusty clips. By properly selecting an optimal set of feature attributes, one can significantly improve the performance of the system. The addition of a radial hole statistics algorithm yields even better results. However, it is not yet possible to meet the requirement for zero false positives with rusty clips on all four clips.

A vision system has been installed on an industrial partner’s inspection line. Significant software modifications have been made to QVision to allow it to work with 4 clips. New feature extraction routines have been created to better analyse the images. Cameras have been chosen and a ring lighting solution has been installed. The working system has been connected to the PLC that is controlling the clipping cell.

In addition to this, new feature selection routines have been employed to be able to improve the performance of the system. For the case of images without rust the system is working properly on the tested datasets.

Field testing has been carried out in order to have the industrial partner sign off on the system. Further research will be presented on new techniques to solve this problem, which will be tested on the library images taken while the cell was in operation.
Chapter 5

Alternate Features and Classification Methods

The results of Chapter 4 have met the criteria of the industrial partner. On a test of over 2000 cross car beams there were 0 FPs and 0.8% FNs, which is better than the goal of 0 FPs and less than 2% FNs. However there is the potential for improvements to this system. The method in Chapter 4 would take a day to find the appropriate feature selection and there was no guarantee that a solution would be found. Additional methods will be investigated in order to make the system not only more robust, but also quicker to train, and possibly eliminate the need for detecting features from the image. The goal is to create a rigorous system that will produce improved results in a systematic way that is also quick to train and hence easily reconfigurable.

The new classification methods to be introduced are: 1) a Neural Network based processor, 2) applying Principal Component Analysis to reclassify the input feature set and 3) using a direct Eigenimage approach to avoid the need to extract features from the image. These methods will be compared in terms of classification accuracy. Additionally new feature detectors will be applied. These are a Hough rectangle extractor, a PCA colour based processing method and a corner detector. These will also be compared to the previous features used in terms of accuracy of classification. An additional data set will also be used to test the performance of these classifiers and new features in detecting orientation defects in addition to presence and absence of clips. The results of these new investigations will be presented showing a comparison of the performance on different datasets.

5.1 New Classification Methods

The ANFIS system based on feature extraction was the basis for the results given in Chapters 3 and 4. An image is acquired and then features are extracted from that image producing numerical inputs. Next a training set of inputs can be created by extracting sets of features from example pass and fail images. A classifier can then be trained based on these sample sets of inputs. Finally, this system can then be used to classify unknown images.
A number of variations on this method will be explored in this chapter. Firstly the classifier can be changed to compare the performance of the ANFIS system to another classifier. In this case a Neural Network classifier has been chosen. Additionally Principal Component Analysis can help reclassify data to provide more separation between pass and fail data sets. Applying PCA on the input data before training both a Neural Network and an ANFIS classifier will be investigated.

In addition to feature based methods it is also possible to classify images directly from their pixel data. This can be accomplished by applying PCA to produce Eigenimages. A Neural Network and an ANFIS system will be trained based on the coefficients of the Eigenimages representing a given image. This enables classification of the images without extracting features.

Table 5.1 summarizes the five new classifiers along with the original feature based ANFIS classifier, which was the basis for the results given in Chapters 3 and 4. The six classifiers are: a) feature based with ANFIS, b) feature based with a Neural Network, c) feature based with PCA applied and ANFIS, d) feature based with PCA applied and a Neural Network, e) Eigenimage based with ANFIS and f) Eigenimage based with a Neural Network. In the table they have been organised into ANFIS vs Neural Network, and Feature Based vs Eigenimage based. By using the six classification methods on the same data set it is possible to compare the results of two intelligent classifiers namely: Neural Networks and ANFIS and to compare the results of feature extraction based image classification with the direct PCA Eigenimage approach.

Table 5.1: Summary of the six classifiers.

<table>
<thead>
<tr>
<th></th>
<th>ANFIS</th>
<th>Neural Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature Based Methods</td>
<td>Feature based with ANFIS</td>
<td>Feature based with a Neural Network</td>
</tr>
<tr>
<td></td>
<td>Feature based with PCA and ANFIS</td>
<td>Feature based with PCA and a Neural Network</td>
</tr>
<tr>
<td>Eigenimage Based Methods</td>
<td>Eigenimage based with ANFIS</td>
<td>Eigenimage based with a Neural Network</td>
</tr>
</tbody>
</table>
5.1.1 Neural Network Classifier

The first new classification method that was added was a Neural Network method. It uses the same input data, obtained by extracting features from the images, as the ANFIS method detailed in Chapter 4. It is also trainable with example data and operates by updating the weights of connections between nodes. A Multi Layer Perceptron (MLP) network was chosen because it commonly appears in literature and is a fairly standard network. The reader is referred to Mehrotra, Mohan and Ranka (2000) for further details on this classifier.

The structure of the network includes one hidden layer. This structure is shown in Figure 5.1. The hidden layer has a sigmoidal activation function and contains 20 nodes. The second layer is the output layer and this only contains one node, with a linear activation function. This node sums the input weights it receives from the hidden layer and outputs that value. The reason 20 nodes were used in the hidden layer was because with 60 feature attributes as inputs into the system, 20 nodes, by means of an informal experiment, showed the best performance. Hypothetically using more hidden nodes could improve the performance of the system, but it then becomes easier to over train the system or over fit the data for the given training set. By using fewer nodes, the chance of over training the network is reduced. A set of checking images is also used to verify that over training is not taking place.

This network was trained by using a backpropagation method. This specific backpropagation technique employs a gradient descent method, with a momentum factor. The moment factor helps avoid local minimum when looking for the global minimum giving the solution momentum in the direction it has been travelling towards (The Mathworks 2008). The network was implemented in the Neural Network Toolbox in MATLAB® using the ‘learngdm’ training method.

![Figure 5.1: Network structure used to train system.](image)
5.1.2 Principal Component Analysis with a Neural Network

One technique for improving the performance of the classification is to apply principal component analysis on the input data prior to processing it (see Kumar 2003). The data is reordered in terms of components with as large a separation between pass and fail sets as possible. Typically only a sample of the new components needs to be used because the first few components provide the highest degree of separation.

The components are generated from the original input features. Only enough components to account for 95% of the original data separation were used. This number was chosen to account for most of the variance, while keeping the number of components low. The number 95% was chosen based on experience. Figure 5.2 illustrates 5 of these components as an example input to the Neural Network classifier. The network is an MLP structure with a gradient descent momentum based backpropagation training technique. The structure is identical to the Neural Network classifier seen in Figure 5.1.

![Figure 5.2: Principal components used to train a Neural Network for the PCA Neural Network classifier. No scales have been provided because the degree of separation is the item of importance in this case.](image)
5.1.3 Principal Component Analysis with ANFIS

Principal component analysis can also be applied to improve the performance of an ANFIS system (for example see Avci and Turkoglu 2009). The same procedure is applied to the input feature data. The data from Figure 5.2 can be input into a classifier. In this case it is an ANFIS network.

An advantage of this approach is a simpler ANFIS network. For example with 60 feature attributes as inputs the network is very complicated. When PCA is applied this can be reduced for example to 5 new features with a high degree of separation. Figure 5.3 illustrates the ANFIS network using the components from Figure 5.2 as inputs.

![Figure 5.3: ANFIS for network with 5 principal components as inputs.](image-url)
5.1.4 Eigenimage based Neural Network Classifier

The Eigenimage PCA based approach provides another way to classify images. It works directly on the pixels of the input images avoiding the need to extract features from an image first. The image is broken down into principal component images or Eigenimages. Only the principal components that make up 99% of the variance of the data are kept. As in Section 5.1.2 this number was chosen to account for most of the variance, while keeping the number of components low. The number 99% was chosen based on experience. Figure 5.4 shows the Eigenimages used for the top left clip. Note these images are the principal component images for both the case when the clip is present and the case when it is not present. Image one contains the largest variations between the pass clips and fails clips. This variation is reduced for each image down to image 14. It can be seen that both image 1 and 2 show a strong outline of the bracket and the hole, whereas images 13 and 14 bear minimal resemblance to the image of the clip and bracket. The principal components after image 14 have been discarded because in total they account for less than 1% of the variance between pass and fail images.

Input images can be broken down into combinations of the Eigenimages. The coefficients that represent this decomposition can then be used as input data to train a classifier (see Ohba and Ilkeuchi 1997). Figure 5.5 shows an example with 14 coefficients used an input to the classifier. Note these coefficients correspond to the Eigenimages in Figure 5.4.

By using these coefficients as inputs the Neural Network classifier can be trained on data. The Neural Network used is the same as before. It is an MLP network with one hidden layer consisting of 20 nodes. It is trained by back propagation using a gradient descent momentum based method.

Murase and Nayer (1995) note that this approach is technically similar to template matching because images with a small distance in Eigenspace have a high correlation. Technically any statistical approach (such as a distance measure) could be used to determine whether the points in Eigenspace belong to a pass image or fail image. The network based approach has been chosen, because it’s trainable.
Figure 5.4: Eigenimages, of the top left clip. These are greyscale images, although the maximum and minimum values may have been truncated to make them visible to the reader.
Figure 5.5: Sample coefficients obtained from Eigenimage decomposition of sample input images. These coefficients correspond to the Eigenimages in Figure 5.4. The upside down triangles represent pass images. The fail triangles represent fail images. The most important feature in this graph is the separation between pass inputs and fail inputs. This is why the horizontal scale has been omitted. The coefficients are used as inputs in the ANFIS and NN classifiers.

One of the options when analysing images using this approach is the preprocessing done on the input image. One of the possible preprocessing steps is to perform edge detections on the image. Figure 5.6 compares the two approaches for a pass clip. Figure 5.7 compares the two approaches for a fail clip. Sobel edge detection was used in this case (see Gonzalez and Woods 2008 for details on Sobel edge detection).
5.1.5 Eigenimage based ANFIS Classifier

The final classifier is an ANFIS based classifier. This classifier is identical to the Eigenimage based Neural Network classifier except that instead of a Neural Network an ANFIS network is trained from the coefficient data.
5.2 New Features

Three new features have been applied to detecting the location of the clip. A generalised Hough transform based rectangle detector has been applied to find the rectangle that represents the outline of the clip. A PCA based colour processing method has been applied to detect the changes in colour between a bracket with the clip present versus a bracket with the clip missing. Finally a corner detector has been applied to detect the corners that are presented because of the clip.

5.2.1 Generalised Hough Transform Rectangle

One of the extensions of the generalised Hough transform (GHT) is using it to find rectangles. The symmetry that rectangles have can be employed to help locate them. First the gradient direction and magnitude of all the pixels are calculated. Say the rectangle has side lengths A and B, where A>B. The sides A are oriented in the direction of the major axis of the rectangle. The sides B are oriented in the direction of the minor axis of the rectangle. Assume that the angle of A is between 0 and 90°, then for a given edge pixel, if its gradient is between 0 and 90° cast votes for a rectangle with a centre on a line at ±B/2 pixels away from the edge pixel in the direction of the edge pixel’s gradient. Alternatively if the directional gradient of the edge pixel is between 90° and 180° then cast votes for a rectangle centred on a line at ±A/2 from the edge pixel.

Then in another plane accumulate votes assuming that the angle of the major axis is between 90° and 180°. If the directional gradient of an edge pixel is between 90° and 180° cast votes for rectangles centred on a line A/2 pixels away from the edge pixel in the direction of the edge pixel’s gradient. Alternatively if the directional gradient of the edge pixel is between 0 and 90° then cast votes for a rectangle centred on a line at ±B/2 from the edge pixel.

This approach will produce two prominent peaks in one of the two accumulators. The largest one is for the major axis and the smaller one is for the minor axis. Where these two peaks intersect should be the centre of the rectangle and the direction should be known because of the slope of the line. Figure 5.8 illustrates this technique. See Davies (2005) for further details.
Figure 5.8: An illustration of the Hough rectangle method. Lines are voted for in two planes. The edges of the grey rectangle are found in both planes. In the one on the left none of the edges will overlap because the orientation is wrong. In the one on the right two sets of edges overlap resulting in a strong identification of the major and minor axis for the rectangle.

Figure 5.9: The Hough rectangle routine locating a clip in the image.
Figure 5.9 demonstrates the Hough rectangle algorithm locating the clip. The rectangular outline of the clip has been identified by this feature detector. The centroid algorithm is also shown because the rectangle is found relative to the centroid.

5.2.2 PCA Colour Feature

To help differentiate differences in colour between an image with the clip present and one without, a PCA based colour approach has been applied. One common form of representing a colour image is based on three colour components, namely: Red, Green and Blue. An image can be broken down into these three images. This is shown in Figure 5.10.

![Figure 5.10: The Red, Green and Blue components of a bracket.](image)
It is possible to apply PCA to these colour channels. The result is three new colour components that have been reclassified in terms of maximum order of intensity Lee et al. (2007). The image is represented as a vector of pixels each with red green and blue values. By applying PCA to this vector new colour components can be generated. Applying SVD (see section 2.4) to the sample correlation matrix will generate eigenvectors and eigenvalues. These eigenvectors can then be used to translate the image into its new colour components. Figure 5.11 shows the new colour components of the bracket from Figure 5.10.

Figure 5.11: The three principal colour components of the bracket. 1) The bracket, 2) the principal colour component with the most variance, 3) the principal colour component with the second most variance, 4) the principal colour component with the least variance, 5) image 3 contrast increased by 31% and brightness by 100% to make the colour components more visible to the user, 6) image 4 contrast increased by 31% and brightness by 100%.
The Eigenvalues of the three components can be used to classify the images because these values will change depending on the presence of certain colours in the image.

5.2.3 Corner Detector

It is possible to detect corners in images. Davies (2005) suggests a median corner detector. The Harris corner detector and a minimum eigenvalue based corner detector (Shi and Tomasi 1994) are other options. The principle behind the median corner detector is to find corners in the image by applying a median filter to the image and then subtracting this result from the original image. The Harris corner detector is also called the Plessey operator and is examined in Nobel (1988). The minimum eigenvalue method is related to the Harris corner detector.

The median corner detector was implemented using an 11x11 median filter on a grey scale input image. The Harris and minimum eigenvalue methods were implemented using MATLAB® library functions for corner detection with the associated default values. The function ‘cornermetric’ was used. Figure 5.12 shows the input pass and fail images. Figure 5.13 displays the results of the median corner detector. Figure 5.14 shows the results of the Harris corner detector. Finally Figure 5.15 shows the results of the minimum eigenvalue corner detector.

The results of the corner detectors were disappointing. Ideally after applying a corner detector points of interest can be identified (i.e. the corners in the image). These can then be used as numerical features based on location, and how strong the corners are. It was hoped that a corner detector could pick up the corners of the rectangular shaped clip, but this has not been the case. The most successful detector was the median detector, but this suffered from a significant amount of noise. This made it difficult to identify the corners of the clip. Neither the Harris detector nor the Minimum Eigenvalue corner detectors were able to successfully identify prominent corners on the edges of the clip. As a result the corner detector was not ultimately applied to detecting the presence of the clips.
Figure 5.12: Input Images.

Figure 5.13: 1) Median corner detector results pass image, 2) median corner detector results fail image, 3) median corner detector results pass image (brightness artificially increased by 75% to be visible to the reader), 4) median corner detector results fail image (brightness artificially increased by 75% to be visible to the reader).
In order to accurately display the essence of the original images Figures 5.14 and 5.15 have been created. First they were scaled down to 50% of their original dimensions in both the x and y direction. Then they were reconverted to binary in order to show the high numbers of potential corners that were found. The resolution of the images has likely been changed moderately through the process of being inserted into this document and printed, however the key features should still be present. These features are namely a very large number of potential corners in most areas of the image except where the radio bracket is located.

Figure 5.14: Harris corner detector (image scaled and converted to binary to retain all the corner points).

Figure 5.15: Eigenvalue corner detector (image scaled and converted to binary to retain all the corner points).
5.3 Results

The new methods introduced above have been implemented into the QVision software package. An additional image set has been added to examine the performance of the system on different lighting and different modes of failure. Tests have been performed on the various datasets to examine the relative performance of the new classifiers and the newly introduced features.

5.3.1 Datasets

In Chapter 4 a total of four data sets were used for testing purposes. For the original set, the training set consisted of 150 pass images and 114 fail images for the top bracket, and 147 pass images and 125 fail images for the bottom bracket. In addition 3 testing sets were collected. The second set consisted of 2016 top images and 2016 bottom images. The third set consisted of 45 top fail images and 60 bottom fail images. For the purposes of the discussion in this chapter the combined second and third testing sets will be referred to as the extended set. The fourth set consisted of 24 top fail and 24 bottom fail rusty images and has not been used in this chapter.

Another additional data set (which will be referred to as the orientation set) has been examined to determine the performance of the system under two conditions. Firstly the lighting has changed in this orientation image set. Secondly 3 new failure modes have also been introduced. Figure 5.16 shows a sample pass image from the orientation set. The clips do not have significant glare, but the background is much more visible in these images. Figures 5.17 through 5.20 show the four methods of failure, which are: missing clip, backwards and upside down clip, backwards clips and upside down clip respectively. The three additional failure modes (upside down and backwards, backwards and lastly upside down clips) could potentially occur in production. They are not however as important to the industrial partner as the missing clips. These failures can be eliminated by other means and the industrial partner has chosen to do that. However, they do make a useful test case.
Figure 5.16: Sample pass clip for the orientation image set.

Figure 5.17: Sample fail (missing clip) for the orientation image set.
Figure 5.18: Sample fail (backwards and upside down clip) for the orientation image set.

Figure 5.19: Sample fail (backwards) for the orientation image set.

Figure 5.20: Sample fail (upside down) for the orientation image set.
5.3.2 Measures of Performance

As in Chapter 4 the measure of performance of the classifiers are based first on the number of false positives, secondly on the number of false negatives and thirdly on the RMS error. ROC curves are also available for classification use.

5.3.3 Effect of Edge Detection on Eigenimage Classification

As has been noted earlier it is possible to perform edge detection on an image before applying the Eigenimage approach. To evaluate the effect of this approach the system was trained using the original dataset from edge based images and the images without edge detection performed. The regions were 200 pixels by 200 pixels found relative to the centroid, which finds the centre of the beam. Figure 5.21 shows an example region of interest chosen. Table 5.2 presents the results of the different classifiers.

Figure 5.21: The region of interest for the top left clip found relative to the centre of the beam.
Table 5.2: Results of edge detection on Eigenimage based classification on the original dataset.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>No Edge Detection Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>9</td>
<td>74</td>
<td>28</td>
<td>0.4182</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>51</td>
<td>91</td>
<td>34</td>
<td>0.4755</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>0.2028</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>6</td>
<td>18</td>
<td>7</td>
<td>0.3166</td>
</tr>
<tr>
<td>1</td>
<td>No Edge Detection Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>0.1836</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>11</td>
<td>4</td>
<td>0.1604</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0798</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0719</td>
</tr>
<tr>
<td>4</td>
<td>Edge Detection Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>19</td>
<td>8</td>
<td>3</td>
<td>0.5088</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>12</td>
<td>7</td>
<td>3</td>
<td>0.7015</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>30</td>
<td>22</td>
<td>8</td>
<td>0.3742</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>33</td>
<td>13</td>
<td>5</td>
<td>0.3250</td>
</tr>
<tr>
<td>2</td>
<td>Edge Detection Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>3</td>
<td>11</td>
<td>4</td>
<td>0.2697</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>11</td>
<td>4</td>
<td>0.2101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>19</td>
<td>7</td>
<td>0.2031</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>15</td>
<td>6</td>
<td>0.2139</td>
</tr>
</tbody>
</table>

As can be seen from Table 5.2 the best classifier was the Neural Network approach with No Edge Detection. This was both in terms of RMS error and in terms of the number of FPs and FNs for all 4 clips. The second best was the Neural Network with Edge Detection in terms of the same criteria. The performance of both the ANFIS networks was poor. The classifiers that did not use edge detection had lower RMS error, but a very high number of false positives for clip 2.

It is recommended that a Neural Network be used without edge detection as this has produced the best results for the Eigenimage based approach.
5.3.4 Classifiers

To test the performance of the 5 new classifiers, the original data set was used to train QVision. Features were defined on the clips. The same set was used for all four feature based techniques (3 new techniques along with the original ANFIS routine). The features used in this are shown and listed in Figure 5.22. Regions of interest, without edge detection, were defined for the two Eigenimage based processing routines. Figure 5.21 shows an image of a typical region of interest. The results of this classification are presented in Table 5.3.

![Figure 5.22: The features used in the first training set.](image)
Table 5.3: Results for the classifiers based on the original dataset.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>1</td>
<td>29</td>
<td>11</td>
<td>0.9587</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>0.3230</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>10</td>
<td>12</td>
<td>4</td>
<td>0.5226</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.2805</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0879</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>2</td>
<td>0.8</td>
<td>0.0899</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0710</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0.0871</td>
</tr>
<tr>
<td>3</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>0</td>
<td>2.3</td>
<td>0.0721</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1102</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>11</td>
<td>4</td>
<td>0.1748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>0.1509</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.0547</td>
</tr>
<tr>
<td>5</td>
<td>Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>2</td>
<td>76</td>
<td>29</td>
<td>0.3932</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>6</td>
<td>40</td>
<td>15</td>
<td>0.3319</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>0.1512</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0.1908</td>
</tr>
<tr>
<td>4</td>
<td>Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>7</td>
<td>3</td>
<td>0.1853</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>0.1166</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0.1304</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.1309</td>
</tr>
</tbody>
</table>
The best results by far were the PCA Neural Network results. Second best were the NN feature based results. The PCA ANFIS results were next best in terms of FPs. The Direct PCA NN followed this then feature based ANFIS and finally Direct PCA ANFIS. This suggests firstly that the Neural Network classifier has better performance than the ANFIS classifier. It also shows that applying PCA improves the performance of the system and that the feature based PCA based approaches perform better than the direct Eigenimage PCA based approaches.

It was desirable to see how the trained solution would stand up to being tested against a much larger test set. The set chosen was the extended set consisting of 2016 top pass images and 2016 bottom pass images along with 45 top fail images and 60 bottom fail images. All of these images are production images from the manufacturing cell at the plant. The classifiers used to generate the data for Table 5.3 were applied to the new testing set. These results are presented in Table 5.4.

Following the pattern from the previous data the PCA Neural Network showed the best results. The NN and PCA ANFIS had similar numbers of false positives and false negatives. The ANFIS classifier was the worst of the feature based classifiers. Neither of the Eigenimage methods produced good results. There were a large number of FPs and FNs for both cases. They all had false positives so none would be suitable for application on the assembly line. Again it is seen that Neural Networks perform better than the ANFIS systems. Applying PCA improves the results and the feature based results are better than the Eigenimage results.

This degradation of the results on a much larger dataset shows that the solution is not robust enough to properly model the much larger dataset. Even the PCA Neural Network approach which had the best results on the original test were not as good when tested on the much larger test set.

In an effort to increase the robustness of the solution new features have been defined. These are the Hough rectangle detector and the PCA colour feature. It is hoped that by adding these features the performance of the feature based classifiers will improve.
Table 5.4: Testing results for extended testing set, trained on the original dataset and tested on the extended data set.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>568</td>
<td>28</td>
<td>1.6459</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>2</td>
<td>348</td>
<td>17</td>
<td>0.4549</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>8</td>
<td>329</td>
<td>16</td>
<td>0.9346</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>7</td>
<td>113</td>
<td>5</td>
<td>0.4179</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>5</td>
<td>18</td>
<td>0.9</td>
<td>0.3497</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>147</td>
<td>7</td>
<td>0.2288</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>120</td>
<td>6</td>
<td>0.1993</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>33</td>
<td>2</td>
<td>0.1351</td>
</tr>
<tr>
<td>3</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>4</td>
<td>24</td>
<td>1</td>
<td>0.1234</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>5</td>
<td>26</td>
<td>1</td>
<td>0.2217</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>344</td>
<td>17</td>
<td>0.2766</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>184</td>
<td>9</td>
<td>0.2372</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>3</td>
<td>23</td>
<td>1</td>
<td>0.0929</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>5</td>
<td>3</td>
<td>0.15</td>
<td>0.0579</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>1</td>
<td>0.05</td>
<td>0.0280</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>44</td>
<td>2</td>
<td>0.1302</td>
</tr>
<tr>
<td>6</td>
<td>Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>6</td>
<td>1570</td>
<td>78</td>
<td>0.6225</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>6</td>
<td>1165</td>
<td>58</td>
<td>0.6200</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>4</td>
<td>130</td>
<td>6</td>
<td>0.2100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>13</td>
<td>141</td>
<td>7</td>
<td>0.1993</td>
</tr>
<tr>
<td>5</td>
<td>Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>17</td>
<td>165</td>
<td>8</td>
<td>0.3223</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>6</td>
<td>382</td>
<td>19</td>
<td>0.3189</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>224</td>
<td>11</td>
<td>0.3044</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>1</td>
<td>70</td>
<td>3</td>
<td>0.0.3957</td>
</tr>
</tbody>
</table>
5.3.5 New Features

The two new features, the Hough rectangle and the PCA colour feature, were tested individually on the original data set in order to determine their success in classifying the presence and absence of the clips. Each of the four clips had the given feature defined relative to the centre of the bracket, which was found using the centroid algorithm with thresholding. All four feature based classifiers were trained with the input features.

Hough Rectangle

Table 5.5 below presents the results of the system trained with the Hough rectangle feature defined on all four clips relative to a centroid.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>31</td>
<td>12</td>
<td>0.2548</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>39</td>
<td>32</td>
<td>12</td>
<td>0.4197</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>6</td>
<td>63</td>
<td>23</td>
<td>0.4312</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>14</td>
<td>65</td>
<td>24</td>
<td>0.4136</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>1</td>
<td>2</td>
<td>0.8</td>
<td>0.1052</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>5</td>
<td>19</td>
<td>7</td>
<td>0.3063</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>2</td>
<td>46</td>
<td>17</td>
<td>0.3401</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>1</td>
<td>13</td>
<td>5</td>
<td>0.2961</td>
</tr>
<tr>
<td>4</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>27</td>
<td>10</td>
<td>0.2304</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>1</td>
<td>59</td>
<td>22</td>
<td>0.3857</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>31</td>
<td>29</td>
<td>11</td>
<td>0.4094</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>27</td>
<td>107</td>
<td>39</td>
<td>0.5051</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>2</td>
<td>0.8</td>
<td>0.1382</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>4</td>
<td>13</td>
<td>5</td>
<td>0.2877</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>14</td>
<td>20</td>
<td>7</td>
<td>0.3849</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>21</td>
<td>8</td>
<td>0.2486</td>
</tr>
</tbody>
</table>
The results of Table 5.5 are not great. There are a large number of FPs and FNs for all four classifiers. In general this feature should not be used alone to classify the clips, but it can contribute to a successful solution. There is one particular area of success. For the PCA Neural Network on Clip 1 there were only 2 FNs and 0 FPs.

**PCA Colour**

Table 5.6 shows the result of the system trained with the PCA Colour feature. The results are very good for both the NN classifier and the PCA based Neural Network classifier. Both have difficulties with Clip 2, but Clips 1, 3 and 4 are classified correctly. Using only this feature produces excellent results on 3 out of the 4 clips with a NN classifier. Note there is no easily identifiable visual difference with Clip 2 to explain this.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>13</td>
<td>5</td>
<td>0.1777</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>25</td>
<td>48</td>
<td>18</td>
<td>0.4244</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>0.1986</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>7</td>
<td>2.6</td>
<td>0.1499</td>
</tr>
<tr>
<td>1</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.1056</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>20</td>
<td>8</td>
<td>0.2862</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.0665</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.366</td>
</tr>
<tr>
<td>4</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>18</td>
<td>7</td>
<td>0.2058</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>22</td>
<td>62</td>
<td>24</td>
<td>0.4419</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>4</td>
<td>4</td>
<td>1.5</td>
<td>0.2953</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>2</td>
<td>35</td>
<td>13</td>
<td>0.2892</td>
</tr>
<tr>
<td>2</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>2</td>
<td>0.8</td>
<td>0.1305</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>22</td>
<td>8</td>
<td>0.4351</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.0606</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0006</td>
</tr>
</tbody>
</table>
5.3.6 New Feature Set

In an effort to increase the performance of the feature based classifiers a new feature set was created. Included in this was the new PCA colour feature and the Hough rectangle feature. Line features and a hole feature were also included along with a centroid feature that all the other features were found relative to. The feature based classifiers were trained on these new features, which were extracted from the original data set. The results of this are presented in Table 5.7.

Table 5.7: Results for new feature set on the original test set.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>0.5047</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.1874</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>5</td>
<td>2</td>
<td>0.1458</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>1</td>
<td>26</td>
<td>10</td>
<td>0.3767</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0488</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.0647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>2</td>
<td>0.7</td>
<td>0.1043</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>1</td>
<td>0.4</td>
<td>0.0908</td>
</tr>
<tr>
<td>3</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>2</td>
<td>0.8</td>
<td>0.0621</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0688</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>12</td>
<td>4</td>
<td>0.1416</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>4</td>
<td>1.5</td>
<td>0.1284</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0047</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>2</td>
<td>0.7</td>
<td>0.0798</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0005</td>
</tr>
</tbody>
</table>

The new feature set was fairly successful. There were only false positives from the ANFIS classifier. The other three classifiers had 0 false positives. Additionally both the Neural Network and PCA Neural Network had very low levels of false negatives (4 and 2 respectively). These are both excellent results. On this new set it is seen that Neural Networks perform better than the ANFIS systems. Applying PCA also improves the results.
To verify that these results were consistent with a larger data set the classifiers were tested on the extended testing set as before. The results of this test are presented in Table 5.8.

Table 5.8 Extended results trained on the original data with the new features and tested on the extended testing set.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>334</td>
<td>16</td>
<td>3.3145</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>16</td>
<td>156</td>
<td>8</td>
<td>0.3500</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>140</td>
<td>7</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>607</td>
<td>30</td>
<td>0.6727</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>207</td>
<td>10</td>
<td>0.2670</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>70</td>
<td>3</td>
<td>0.1766</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>127</td>
<td>6</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>52</td>
<td>3</td>
<td>0.1780</td>
</tr>
<tr>
<td>3</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>0</td>
<td>50</td>
<td>2</td>
<td>0.1220</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>1</td>
<td>5</td>
<td>0.2</td>
<td>0.0787</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>204</td>
<td>10</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>203</td>
<td>10</td>
<td>0.2488</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>0</td>
<td>34</td>
<td>1.7</td>
<td>0.1143</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>368</td>
<td>18</td>
<td>0.3987</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>6</td>
<td>0.3</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>2</td>
<td>0.1</td>
<td>0.0265</td>
</tr>
</tbody>
</table>

These results are better than the results from the first feature set (see Table 5.4). There are significantly fewer FPs in general. The Neural Network classifier and the PCA Neural Network classifier both have zero false positives. TheFNs have also been reduced. The results for clip 3 and 4 for the PCA Neural Network are excellent with very few FNs in addition to zero FPs. Although they are not perfect, these are an improvement on the result from Table 5.4.
5.3.7 Orientation

The final set of images that was examined was the new orientation set. There are four additional modes of failure and the lighting has changed. These additional modes of failure can be caught by other methods and the industrial partner does not need the vision system to inspect for these defects. However it does present an excellent set of data to test the robustness of the algorithm. As in the previous cases features were defined on the images. However these were not found relative to the centre of the beam because the centroid feature did not work under the new lighting. The feature based results are presented in Table 5.9.

Table 5.9: Results for feature based approach on orientation test set.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>ANFIS</td>
<td>Clip 1</td>
<td>11</td>
<td>9</td>
<td>4</td>
<td>2.1610</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>0.1999</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>22</td>
<td>18</td>
<td>8</td>
<td>1.9781</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>32</td>
<td>31</td>
<td>15</td>
<td>6.5963</td>
</tr>
<tr>
<td>2</td>
<td>Neural Network</td>
<td>Clip 1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.1326</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.1202</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>8</td>
<td>10</td>
<td>4</td>
<td>0.3118</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>0.1961</td>
</tr>
<tr>
<td>3</td>
<td>PCA ANFIS</td>
<td>Clip 1</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>0.2373</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>3</td>
<td>27</td>
<td>12</td>
<td>0.2599</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>4</td>
<td>16</td>
<td>8</td>
<td>0.2946</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>11</td>
<td>23</td>
<td>11</td>
<td>0.3106</td>
</tr>
<tr>
<td>1</td>
<td>PCA Neural Network</td>
<td>Clip 1</td>
<td>2</td>
<td>1</td>
<td>0.5</td>
<td>0.1681</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0.1922</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>0.2543</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>6</td>
<td>7</td>
<td>3</td>
<td>0.2544</td>
</tr>
</tbody>
</table>

All the classifiers produced FPs, which is unacceptable. There are also a significant number of FNs for all classifiers. There results are not good enough for use in an industrial machine vision
It is important to note that again the performance of the Neural Network classifier is better than the performance of the ANFIS classifier. Additionally applying PCA also improves the results.

The Eigenimage classifiers were also investigated. Regions of interest were defined as in Figure 5.21. Edge detection was considered again because of the difference in the image sets. These results are presented in Table 5.10.

### Table 5.10: Results for Eigenimage approach on orientation test set.

<table>
<thead>
<tr>
<th>Rate</th>
<th>Classifier</th>
<th>Clip</th>
<th>False Positives</th>
<th>False Negatives</th>
<th>% False Negatives</th>
<th>RMS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>No Edge Detection Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>12</td>
<td>2</td>
<td>0.9</td>
<td>0.2253</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>2</td>
<td>5</td>
<td>2.29</td>
<td>0.1626</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>0.1228</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>1</td>
<td>2</td>
<td>0.9</td>
<td>0.0963</td>
</tr>
<tr>
<td>1</td>
<td>No Edge Detection Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0.1835</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0662</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0646</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0548</td>
</tr>
<tr>
<td>4</td>
<td>Edge Detection Direct PCA ANFIS</td>
<td>Clip 1</td>
<td>5</td>
<td>10</td>
<td>5</td>
<td>2.9485</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>15</td>
<td>9</td>
<td>4</td>
<td>1.8040</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>2</td>
<td>16</td>
<td>8</td>
<td>1.1873</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>1.3885</td>
</tr>
<tr>
<td>2</td>
<td>Edge Detection Direct PCA Neural Network</td>
<td>Clip 1</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0.2269</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 2</td>
<td>3</td>
<td>2</td>
<td>0.9</td>
<td>0.1783</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 3</td>
<td>0</td>
<td>2</td>
<td>0.9</td>
<td>0.1660</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clip 4</td>
<td>6</td>
<td>2</td>
<td>0.9</td>
<td>0.1829</td>
</tr>
</tbody>
</table>

The best classifier was the Eigenimage PCA Neural Network with no edge detection which had no false negatives. Clip 1 has 9 false positives, which is significant but Clips 2 through 4 had no false positives. The Eigenimage PCA ANFIS technique and the Eigenimage PCA Neural Network technique with edge detection both had similar results and the Eigenimage PCA ANFIS
classifier with edge detection had the worst results. As seen before the results without edge
detection are better than the results with edge detection. Additionally the Neural Network
showed better performance than the ANFIS classifier.

Of particular interest is the direct PCA Neural Network classifier without using edge detection.
Ignoring Clip 1 these are perfect results and are even better than the feature based results.

5.3.8 Discussion

Six different classifiers were compared. Tests were done on three image sets and with two
different feature sets. Consistently it was seen that a Neural Network classifier whether used on
feature data, on feature data with PCA applied or on Eigenimage coefficients performed better
than the ANFIS system. The NN results always ranked higher on all the tests. It is reasonable to
say that the MLP Neural Network with one hidden layer performs better than the ANFIS system.
It has also been consistently seen that applying PCA on input data improves the results of
classification. The results with feature extraction and PCA ranked higher that the results with
feature extraction and no PCA on the majority of the tests.

It is not initially apparent why the NN performed better than the ANFIS system. However since
the ANFIS system is a fuzzy system it might have trouble making crisp distinctions between pass
and fail images. It is understandable that when PCA is applied the results improved. The system
is smaller and simpler with inputs being reclassified for maximized variance between pass and
fail sets. This should make it easier to classify the data properly.

The Eigenimage results may not have performed as well as the feature based results because the
part had a +/- 2mm tolerance for its location. The changing location may have made it difficult to
use the Eigenimage technique, even though the regions of interest were found using the centroid
technique to locate the centre of the beam. For the feature based techniques some feature
extractors may be more robust to changes in the bracket’s location. For example for some
extractors the strength of the line is not directly dependent on the exact location of that line.

It is not surprising that edge detection did not improve the Eigenimage results, as this approach
has not seen in the literature.
For the original data set and the extended testing set the performance of feature based techniques were better than region of interest techniques. In the case of the orientation data set in general the feature based techniques had similar performance to the Eigenimage techniques. However the Neural Network technique with no edge detection worked better than any other technique on this set of data. It was also seen that using the edge detection methods did not improve the Eigenimage techniques.

Unfortunately the excellent results (0 FPs and less than 2% FNs) especially by the PCA Neural Network classifier on the first data set did not always hold when testing the extended data set. This was seen in several cases for the new feature set although there were excellent results for Clips 3 and 4 of the PCA Neural Network classifier for the new feature set. It is plausible that the features used may not be robust enough to accurately model the new set of data from the given inputs. However this is unlikely because even with the introduction of two new features -- the Hough rectangle and the PCA colour feature -- this wasn’t completely solved. Alternatively the original training set itself may not be an accurate description of the much larger data set.

Although there were difficulties in extending an initial training set to an extended testing set there was some success. In contrast to the results from Chapter 4 these new classifiers and features are quickly trained. The feature selection process from Chapter 4 would typically need to run overnight, whereas all the new methods can be trained in minutes.

The feature selection techniques from Chapter 4 are computationally intensive and are not guaranteed to produce good results. However applying PCA to a dataset eliminates the need to perform feature selection improving the result in a systematic way. There is also no guarantee even if feature selection is used the results will extend to a larger data set. The Eigenimage technique has the benefit of not needing to extract features. A region of interest is selected and the calculations can then proceed. The greatest benefit of these new techniques is their speed of training, which makes the system more flexible.
5.4 Summary

The system in Chapter 4 was improved upon by investigating additional methods that are more robust and quicker to train. One method (the Eigenimage approach) even eliminated the need for detecting features from the image. A rigorous system was created that produced results in a systematic way. The results were not significantly improved. However the training time was reduced from hours for the original method to just minutes.

Five new classifiers were introduced. Three of these were feature based, requiring features to be extracted from the image before processing. The first of the three was a Neural Network classifier. The second two involved applying PCA to the data first and then classifying it using an ANFIS classifier or a Neural Network. There were two Eigenimage classifiers investigated, which used the coefficients of Eigenimage decomposition as inputs to either an ANFIS system or a Neural Network. Additionally three new feature detectors were introduced. These were the Hough rectangle, a PCA colour based feature and a corner detector. The corner detector did not shown promise, but both the Hough rectangle detector and the PCA colour feature were investigated.

These classifiers and features were tested on three main data sets. The first is the original set, the second is an extended training set with 2016 pass images and 45 top fail images and 60 bottom fail images. The last is the orientation data set with 4 new methods of failure. Results with no FPs and less than 2% FNs were achieved on the first data set, but these did not hold consistently on the extended set even with the use of new features. The feature based approaches did not have much success on the orientation data set, but the Neural Network based Eigenimage classifier with no edge detection displayed perfect results for Clips 2, 3 and 4.

In conclusion it was found that the Neural Network classifier performed better than the ANFIS classifier. When PCA was applied the results improved. It was better not to use edge detection to preprocess regions of interest for the Eigenimage based methods. Finally, overall, feature based methods had better performance than Eigenimage based methods. Although the classification accuracy was only slightly improved compared, to Chapter 4, the system can be trained in minutes instead of hours. This provides a superior MV system.
Chapter 6

Conclusions and Recommendations

6.1 Conclusions

A working machine vision system was successfully installed on an industrial assembly line. The system was used to verify the presence of four J-clips on the radio bracket of a cross car beam. Two different approaches for vision based inspection were examined. The first was a feature based method, where features such as lines, holes and colour based features were extracted from an image. The image was then evaluated based on the parameters of these features. To improve the classification results, feature selection could be applied or principal component analysis could be used. The second approach considered the entire image as a matrix and processed it as an Eigenimage using linear algebra techniques to redefine the data as a series of coefficients of component images that make up the image. The image was then evaluated based on this data. These methods were evaluated to determine which one was better. It was found that the Neural Network method performed better than the ANFIS method. When PCA was applied the results improved still further. Overall, feature based methods had better performance than Eigenimage based methods. Finally, it should be noted that these methods required only minutes to train.

6.1.1 Laboratory Test Cell

In Chapter 3 it was found that the system performance is better when the camera auto-adjust settings were turned on. For the networks where feature selection was not used there was a large increase in performance from Trial 1, with the auto-adjust settings off, to Trial 3 with the auto-adjust settings on. The system also performed significantly better when feature selection was used. In both cases by using feature selection uncertainty in classification was reduced and the RMS error was drastically reduced. Thus, the optimal system is with the camera auto-adjust on and the network optimized. The RMS error for the optimal case was practically speaking zero.
Note that both trial 2 and trial 4 showed very good results. The reason to use the auto-adjust results (trial 4) is because of robustness of the system. The RMS error is lower in trial 4 and the results are better classified. Note also that the shutter settings were manually adjusted for trial 2. Using the camera auto-adjust settings avoids the need to tune this parameter.

A laboratory test cell was constructed to test a variety of lighting solutions for J-clip detection on the radio bracket of a cross car beam. Fluorescent lights, LED spotlights and LED rings lights were used as light sources. A CCD camera with optional auto settings was used to acquire images of the radio bracket.

QVision is a GUI based image inspection software program. By extracting lines, holes, circles and colour features from an image an ANFIS system can be trained. This system can then be used to classify live images. Tests were conducted using the various lighting setups, and the various camera settings. Feature selection was also used to improve the performance of the ANFIS system. When the camera auto settings were used the classification results were more accurate. When feature selection was used the classification results were also more accurate. When both the camera auto settings and feature selection were used the best results were obtained.

It is recommended that the lighting stay constant. Cameras with auto settings can be used to help achieve this. By employing feature selection it is also possible to increase the accuracy of classification results.

### 6.1.2 Industrial Manufacturing Cell

In Chapter 4 it was found that it is possible to generate acceptable results on the given data set without rusty clips. By properly selecting an optimal set of feature attributes, one can significantly improve the performance of the system. The addition of a radial hole algorithm yielded even better results. However, it is not yet possible to meet the requirement for zero false positives with rusty clips.

A vision system was installed on an industrial partner’s inspection line. Significant software modifications were made to QVision to allow it to work with 4 clips. New feature extraction
routines were created to better analyse the images. Cameras were chosen and a ring lighting solution was installed. The working system was connected to the PLC controlling the clipping cell.

In addition to this, new feature selection routines were employed to improve the performance of the system. For the case of images without rust the system worked properly on the tested datasets. However it is not currently possible to eliminate all false positives when the case of the rusty bracket is investigated. Finally field testing was carried out in order to have the industrial partner sign off on the system.

6.1.3 Alternate Features and Classification Methods

In Chapter 5 the system in Chapter 4 was improved upon by investigating additional methods that are more robust and quicker to train. One method (the Eigenimage approach) even eliminated the need for detecting features from the image. A rigorous system was created that improved results in a systematic way. The accuracy of the classification results was marginally better. However the time it took for training was reduced from hours for the original method to just minutes.

Five new classifiers were introduced. Three of these were feature based, requiring features to be extracted from the image before processing. The first of the three was a Neural Network classifier. The second two involved applying PCA to the data first and then classifying it using an ANFIS classifier or a Neural Network. There were two Eigenimage classifiers investigated, which used the coefficients of Eigenimage decomposition as inputs to either an ANFIS system or a Neural Network. Additionally three new feature detectors were introduced. These were the Hough rectangle, a PCA colour based feature and a corner detector. The corner detector did not shown promise, but both the Hough rectangle detector and the PCA colour feature were investigated.

These classifiers and features were tested on three main data sets. The first is the original set, the second is an extended training set with 2016 pass images and 45 top fail images and 60 bottom fail images. The last is the orientation data set with 4 new methods of failure. Results with no FPs and less than 2% FNIs were achieved on the first data set, but these did not hold consistently
on the extended set even with the use of new features. The feature based approaches did not have much success on the orientation data set, but the Neural Network based Eigenimage classifier with no edge detection displayed perfect results for Clips 2, 3 and 4.

In conclusion it was found that the Neural Network classifier performed better than the ANFIS classifier. When PCA was applied the results improved. It was better not to use edge detection to preprocess regions of interest for the Eigenimage based methods. Finally overall feature based methods had better performance than Eigenimage based methods. Although the classification accuracy was only slightly improved, compared to Chapter 4, the system can be trained in minutes instead of hours. This provides a superior MV system.

6.2 Recommendations

There are five recommendations with regards to the future direction to this research. The first recommendation is to apply the developed techniques to a new test case. This will provide an opportunity to prove the flexibility and the robustness of the system. It require a new lighting setup and a new camera setup on a new assembly line, however it is the only way to verify the flexibility of the system based on other industrial inspection tasks.

A second area of further research would be to implement the approach of Sun, Sun and Surgenor, (2008) to compare an additional method of implementing the Eigenimage approach. The Eigenimage approach holds significant promise and further investigation could be very useful.

The RANSAC method (Fischler and Bolles 1981) presents an alternative to the Hough transform for finding lines circles in images. It would be interesting to see how the RANSAC method affects the performance of the system especially in comparison to the Hough transform results.

The fourth and fifth recommendations for future work are detailed in the following section. The third is a statistical modeling approach to provide statistical based results on the expected performance of the system based on the nature of the system itself as opposed to only the number of FPs and FNs in a given training set. The fourth recommendation is to train a system based on a 3D model of the part to allow for the training of a machine vision system even before the manufacturing cell has been commissioned.
6.2.1 Statistical Modeling

One of the difficulties with the approaches presented so far is in dealing with false positives. Luko (2000) talks about the practicalities of achieving “Zero Defects” from a Statistical standpoint. Luko points out the example of a set of 10000 parts where a defect is either present or not. If 1000 parts are sampled at random and zero defects are found in that batch the probability of having zero defects in the entire set is only 10%. It would take 9000 parts to be sampled (all having zero defects) to have a 90% certainty that there were 0 defects in the 10000 parts.

This same logic can be applied to extrapolating zero false positives. Considering the vision system as a black box and 100 defective classified correctly. Based on these results the probability that there are zero false positives in the next 100 defective parts has an upper bound of 50%. The system may perform much better, but statistically speaking there is no certainty of that.

It would be very useful to have a stronger model for predicting the accuracy of the system than one simply based on the number of false positives and the number of false negatives output by the system. This means moving away from a measure of performance based simply on outputs and considering the function of the system based on given inputs (i.e. the feature data of the coefficient of the Eigenimages). If the system is viewed as a mathematical operation working on input data, given certain inputs certain outputs will result. Therefore if certain probabilities can be assigned to given inputs it should be possible to assign certain probabilities to certain outputs.

Both the ANFIS system and the Neural Network system can be modeled mathematically as a function once they are trained. Large datasets (2000 + images) are available to define input probabilities, hence it should be possible to propagate these probabilities through the classifier and generate an output probability density function based on these input probabilities. This would allow for stronger statements to be made about the performance of the system based on the input probabilities of the individual features having certain values.
6.2.2 Training from a 3D Part Model

Another area of future work is the ability to train the system from a 3D model of the part. This would provide the ability to configure the system earlier on in the development phase of the manufacturing process hence it is highly desirable. Features could be extracted from a 2D rendering of the 3D part model. This could be done for both the case of a pass clip and the case of a fail clip. These features from the 3D model could then be matched to features of the actual images of the clips.

As noted by Zitová and Flusser (2003) it is possible to register multimodal images based on features and this is often done for medical imaging purposes. The projection of the 3D model of the part and the actual images of the part may share common features. By extracting features that are not very sensitive to lighting changes and the surfaces of the part (i.e. geometric features as opposed to colour based features) from the 3D model of the part it may be possible to perform initial training of the system from the model.

This would be a difficult task to perform because of the differences in the 3D model and the actual part. However registration can be performed between MR images and CT images so similar techniques might be successfully applied to this problem.
References


Miles, B.C. and Surgenor, B.W., (2009), “Industrial experience with a machine vision system for the detection of missing clips,” *Changeable, Agile, Reconfigurable and Virtual Production (CARV 2009)*, Munich, Germany, October 5-7th.


Appendix A

Camera and Lighting Specifications

A.1 Camera Specifications

The camera used was an IMI-TECH 1080 FT camera. It has an auto exposure control, which controls the shutter / gain using a feedback loop based on the average pixel luminance. There is also a white balance control, which alters the ratio of Red to Green pixel values and Green to Blue pixel values. This can be adjusted automatically. The reader is referred to the user manual for this camera for further details (see imi-tech, 2009).

A.2 Lighting Specifications

The fluorescent light was a “Model 13 High Output Fluorescent Linear High Frequency (25 kHz) Lite Mite™ Series.” For further reference the reader is referred to Stocker Yale (2009).

The LED spotlight was a NERLITE® S-40 Series Spot Array (NER S-40, W LED-ND, 12V-C). For further reference the reader is referred to Microscan Systems (2009).

The Ring light was a NERLITE® R-100-2 “V2” Series Ring light (R-100-2, V2 W LED-D, 12V-C). For further reference the reader is referred to Microscan Systems (2009).
Appendix B

Dataset Description

This appendix contains a description of the features used in the training sets for Chapters 4 and 5.

B.1 Features used for ‘No Rust Dataset’ in Chapter 4

Table B.1 contains the features used in feature selection 2 from Table 4.1. Table B.2 contains the Features used in ‘No Rust’ from Table 4.2. Note the for tables B.1 and B.2 the same features are used with slightly different performance. This was because the training and checking image sets, for training purposes, were slightly different.

Table B.1: Features used in Feature Selection 2 from Table 4.1

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features Used</th>
<th>Clip</th>
<th>Features Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td>Line Right – Score</td>
<td>Clip 2</td>
<td>Line Right – Score</td>
</tr>
<tr>
<td></td>
<td>Circle – Radius</td>
<td></td>
<td>Hole – X</td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td>Hole – Score</td>
</tr>
<tr>
<td></td>
<td>Centroid Threshold – X</td>
<td></td>
<td>Colour Stats – Red</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clip 2</td>
<td>Line Right – Score</td>
<td>Clip 4</td>
<td>Line Right – Score</td>
</tr>
<tr>
<td></td>
<td>Hole – X</td>
<td></td>
<td>Hole – Score</td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td>Colour Stats – Red</td>
</tr>
<tr>
<td></td>
<td>Colour Stats – Red</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clip 3</td>
<td>Line Right – Score</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Colour – Hue</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Colour – Area</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B.2: Features used in ‘No Rust’ from Table 4.2

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features Used</th>
<th>Clip</th>
<th>Features Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td>Line Right – Score</td>
<td>Clip 2</td>
<td>Line Right – Score</td>
</tr>
<tr>
<td></td>
<td>Circle – Radius</td>
<td></td>
<td>Hole – X</td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td>Hole – Score</td>
</tr>
<tr>
<td></td>
<td>Centroid – X</td>
<td></td>
<td>Colour Stats – Red</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clip 2</td>
<td>Line Right – Score</td>
<td>Clip 4</td>
<td>Line Right – Score</td>
</tr>
<tr>
<td></td>
<td>Hole – X</td>
<td></td>
<td>Hole – Score</td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td>Colour Stats – Red</td>
</tr>
<tr>
<td></td>
<td>Colour Stats – Red</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clip 3</td>
<td>Line Right – Score</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hole – Score</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Colour – Hue</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Colour – Area</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table B.3 contains the features used in both Feature Selection 1 from Table 4.1 and in the ‘No Rust’ unoptimised dataset. Images of these features on both pass and fail clips are shown in figures B.1 through B.8.

Table B.3: Features used in ‘NoRust’ unoptimised dataset and in Feature Selection 1 from Table 4.1 in Chapter 4.

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features</th>
<th>Clip</th>
<th>Features</th>
</tr>
</thead>
</table>
| Clip 1 | Line – Left  
Line – Right  
Line – Top  
Line – Bottom  
Circle  
SemiCircle  
Colour  
Hole  
Line Diagonal – Left  
Line Diagonal – Right  
Centroid – Threshold  
Colour Stats  
Centroid – Grayscale  
Hole Stats – $R_w = 2$  
Hole Stats – $R_w = 4$ | Clip 2 | Line – Right  
Line – Left  
Line – Top  
Line – Bottom  
Circle  
Hole  
SemiCircle  
Colour  
Line Diagonal – Right  
Line Diagonal – Left  
Colour Stats  
Centroid – Threshold  
Centroid – Grayscale |
| Clip 3 | Line – Left  
Line – Right  
Line – Top  
Line – Bottom  
Circle  
Hole  
SemiCircle  
Colour  
Line Diagonal – Right  
Line Diagonal – Left  
Colour Stats  
Centroid – Threshold  
Centroid – Grayscale | Clip 4 | Line – Left  
Line – Right  
Line – Top  
Line – Bottom  
Circle  
Hole  
SemiCircle  
Colour  
Line Diagonal – Right  
Line Diagonal – Left  
Colour Stats  
Centroid – Threshold  
Centroid – Grayscale |
Figure B.1: Top left pass clip. Features used in ‘No Rust’ dataset in Chapter 4.

Figure B.2: Top left fail clip. Features used in ‘No Rust’ dataset in Chapter 4.
Figure B.3: Top right pass clip. Features used in ‘No Rust’ dataset in Chapter 4.

Figure B.4: Top right fail clip. Features used in ‘No Rust’ dataset in Chapter 4.
Figure B.5: Bottom left pass clip. Features used in ‘No Rust’ dataset in Chapter 4.

Figure B.6: Bottom left fail clip. Features used in ‘No Rust’ dataset in Chapter 4.
Figure B.7: Bottom right pass clip. Features used in ‘No Rust’ dataset in Chapter 4.

Figure B.8: Bottom right fail clip. Features used in ‘No Rust’ dataset in Chapter 4.
B.2 Features used for ‘Original’ dataset in Chapter 5

The features used in the ‘Original’ dataset for Chapter 5 are presented. Table B.4 details these features and Figures B.9 through B.16 show these features defined on both pass clip images and fail clip images.

Table B.4: Features used in ‘Original’ dataset in Chapter 5.

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features</th>
<th>Clip</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td>Line – Left&lt;br&gt;Line – Right&lt;br&gt;Line – Top&lt;br&gt;Line – Bottom&lt;br&gt;Circle&lt;br&gt;SemiCircle&lt;br&gt;Colour&lt;br&gt;Hole&lt;br&gt;Line Diagonal – Right&lt;br&gt;Line Diagonal – Left&lt;br&gt;Centroid – Threshold&lt;br&gt;Colour Stats&lt;br&gt;Centroid – Grayscale</td>
<td>Clip 2</td>
<td>Line – Left&lt;br&gt;Line – Right&lt;br&gt;Line – Top&lt;br&gt;Line – Bottom&lt;br&gt;Circle&lt;br&gt;Hole&lt;br&gt;SemiCircle&lt;br&gt;Colour&lt;br&gt;Line Diagonal – Right&lt;br&gt;Line Diagonal – Left&lt;br&gt;Colour Stats&lt;br&gt;Centroid – Threshold&lt;br&gt;Centroid – Grayscale</td>
</tr>
<tr>
<td>Clip 3</td>
<td>Line – Right&lt;br&gt;Line – Left&lt;br&gt;Line – Top&lt;br&gt;Line – Bottom&lt;br&gt;Circle&lt;br&gt;Hole&lt;br&gt;SemiCircle&lt;br&gt;Colour&lt;br&gt;Line Diagonal – Right&lt;br&gt;Line Diagonal – Left&lt;br&gt;Colour Stats&lt;br&gt;Centroid – Threshold&lt;br&gt;Centroid – Grayscale</td>
<td>Clip 4</td>
<td>Line – Left&lt;br&gt;Line – Right&lt;br&gt;Line – Top&lt;br&gt;Line – Bottom&lt;br&gt;Circle&lt;br&gt;Hole&lt;br&gt;SemiCircle&lt;br&gt;Colour&lt;br&gt;Line Diagonal – Right&lt;br&gt;Line Diagonal – Left&lt;br&gt;Colour Stats&lt;br&gt;Centroid – Threshold&lt;br&gt;Centroid – Grayscale</td>
</tr>
</tbody>
</table>
Figure B.9: Top left pass clip. Features used in ‘Original’ dataset in Chapter 5.

Figure B.10: Top left fail clip. Features used in ‘Original’ dataset in Chapter 5.
Figure B.11: Top right pass clip. Features used in ‘Original’ dataset in Chapter 5.

Figure B.12: Top right fail clip. Features used in ‘Original’ dataset in Chapter 5.
Figure B.13: Bottom left pass clip. Features used in ‘Original’ dataset in Chapter 5.

Figure B.14: Bottom left fail clip. Features used in ‘Original’ dataset in Chapter 5.
Figure B.15: Bottom right pass clip. Features used in ‘Original’ dataset in Chapter 5.

Figure B.16: Bottom right fail clip. Features used in ‘Original’ dataset in Chapter 5.
**B.3 Features used for ‘New’ dataset in Chapter 5**

The features used in the ‘New’ feature set trained on the original dataset for Chapter 5 are presented (see section 5.3.6). Table B.5 details these features and Figures B.17 through B.24 show these features defined on both pass clip images and fail clip images.

Table B.5: Features used in ‘New’ dataset in Chapter 5.

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features</th>
<th>Clip</th>
<th>Features</th>
</tr>
</thead>
</table>
Figure B.17: Top left pass clip. Features used in ‘New’ dataset in Chapter 5.

Figure B.18: Top left fail clip. Features used in ‘New’ dataset in Chapter 5.
Figure B.19: Top right pass clip. Features used in ‘New’ dataset in Chapter 5.

Figure B.20: Top right fail clip. Features used in ‘New’ dataset in Chapter 5.
Figure B.21: Bottom left pass clip. Features used in ‘New’ dataset in Chapter 5.

Figure B.22: Bottom left fail clip. Features used in ‘New’ dataset in Chapter 5.
Figure B.23: Bottom right pass clip. Features used in ‘New’ dataset in Chapter 5.

Figure B.24: Bottom right fail clip. Features used in ‘New’ dataset in Chapter 5.
B.4 Features used for ‘Orientation’ dataset in Chapter 5

The features used in the ‘Orientation’ dataset for Chapter 5 are presented. Table B.6 details these features and Figures B.25 through B.32 show these features defined on both pass clip images and fail clip images.

Table B.6: Features used in ‘Orientation’ dataset in Chapter 5.

<table>
<thead>
<tr>
<th>Clip</th>
<th>Features</th>
<th>Clip</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clip 1</td>
<td>Line – Left</td>
<td>Clip 2</td>
<td>Line – Left</td>
</tr>
<tr>
<td></td>
<td>Line – Right</td>
<td></td>
<td>Line – Right</td>
</tr>
<tr>
<td></td>
<td>Line – Bottom</td>
<td></td>
<td>Line – Bottom</td>
</tr>
<tr>
<td></td>
<td>Line – Top</td>
<td></td>
<td>Line – Top</td>
</tr>
<tr>
<td></td>
<td>Circle</td>
<td></td>
<td>Circle</td>
</tr>
<tr>
<td></td>
<td>SemiCircle</td>
<td></td>
<td>Hole</td>
</tr>
<tr>
<td></td>
<td>Colour</td>
<td></td>
<td>SemiCircle</td>
</tr>
<tr>
<td></td>
<td>Hole</td>
<td></td>
<td>Colour</td>
</tr>
<tr>
<td></td>
<td>Line Diagonal – Right</td>
<td></td>
<td>Line Diagonal – Right</td>
</tr>
<tr>
<td></td>
<td>Line Diagonal – Left</td>
<td></td>
<td>Line Diagonal – Left</td>
</tr>
<tr>
<td></td>
<td>Centroid – Threshold</td>
<td></td>
<td>Colour Stats</td>
</tr>
<tr>
<td></td>
<td>Colour Stats</td>
<td></td>
<td>Centroid – Threshold</td>
</tr>
<tr>
<td></td>
<td>PCA Colour</td>
<td></td>
<td>PCA Colour</td>
</tr>
<tr>
<td>Clip 3</td>
<td>Line – Left</td>
<td>Clip 4</td>
<td>Line – Left</td>
</tr>
<tr>
<td></td>
<td>Line – Right</td>
<td></td>
<td>Line – Right</td>
</tr>
<tr>
<td></td>
<td>Line – Bottom</td>
<td></td>
<td>Line – Bottom</td>
</tr>
<tr>
<td></td>
<td>Line – Top</td>
<td></td>
<td>Line – Top</td>
</tr>
<tr>
<td></td>
<td>Circle</td>
<td></td>
<td>Circle</td>
</tr>
<tr>
<td></td>
<td>Hole</td>
<td></td>
<td>Hole</td>
</tr>
<tr>
<td></td>
<td>SemiCircle</td>
<td></td>
<td>SemiCircle</td>
</tr>
<tr>
<td></td>
<td>Colour</td>
<td></td>
<td>Colour</td>
</tr>
<tr>
<td></td>
<td>Line Diagonal – Right</td>
<td></td>
<td>Line Diagonal – Right</td>
</tr>
<tr>
<td></td>
<td>Line Diagonal – Left</td>
<td></td>
<td>Line Diagonal – Left</td>
</tr>
<tr>
<td></td>
<td>Colour Stats</td>
<td></td>
<td>Colour Stats</td>
</tr>
<tr>
<td></td>
<td>Centroid – Threshold</td>
<td></td>
<td>Centroid – Threshold</td>
</tr>
<tr>
<td></td>
<td>PCA Colour</td>
<td></td>
<td>PCA Colour</td>
</tr>
</tbody>
</table>

Figure B.25: Top left pass clip. Features used in ‘Orientation’ dataset in Chapter 5.

Figure B.26: Top left fail clip. Features used in ‘Orientation’ dataset in Chapter 5.
Figure B.27: Top right pass clip. Features used in ‘Orientation’ dataset in Chapter 5.

Figure B.28: Top right fail clip. Features used in ‘Orientation’ dataset in Chapter 5.
Figure B.29: Bottom left pass clip. Features used in ‘Orientation’ dataset in Chapter 5.

Figure B.30: Bottom left fail clip. Features used in ‘Orientation’ dataset in Chapter 5.
Figure B.31: Bottom right pass clip. Features used in ‘Orientation’ dataset in Chapter 5.

Figure B.32: Bottom right fail clip. Features used in ‘Orientation’ dataset in Chapter 5.
Appendix C

Algorithm Details and Code Listings

This appendix contains a listing of the important parts of the computer vision algorithm. Subroutines are listed where available. The reader is referred to the MATLAB® documentation for any MATLAB® specific code, whereas a brief explanation is provided for any custom software.

C.1 Summary of Algorithms

The components of the algorithm are the feature extraction routines, the classifications algorithms, feature selection and performance evaluation techniques. These are presented in corresponding sections in this appendix. An additional area of note is subroutines that are from outside sources. These are not part of the MATLAB® software and have not been written by either Killing (2007) or by Miles. These are cited by author.

C.2 Feature Extraction Algorithms

The following are the feature extraction routines and the region of interest (ROI) area extraction routines. If multiple functions are used they are all listed.

C.2.1 Geometric and Colour Features

Line

\[
function \ line = \text{Line\_Hough\_Grd(img, mskimg,thetain,rhoin)}
\]

Computes the Hough transform of a greyscale image without converting to a binary image (Peng 2006).

img – input image
mskimg – mask of the area to use as the input
thetain – the theta input for the line
rhoin – the rho input for the line
line – the x1, x2 and y1 and y2 coordinates of the line segment

function [fVect, limits] = LineDetectHough(srcImage, X, Y, Width, Height, AngleMin, AngleMax, Polarity, HoG)

Basic line detection routine

srcImage – the input source image
X – the X coordinate for the area of interest to look for the line in
Y – the Y coordinate for the area of interest to look for the line in
Width – the Width of the area of interest
Height – the Height of the area of interest
AngleMin – the minimum angle of the line
AngleMax – the maximum angle of the line
Polarity - unused
HoG – whether to use Histogram of Gradients Prefiltering

fVect – a vector containing the theta, rho, strength and length of the line
limits – numerical limits on fVect

function [varargout] = Hough_Grd(img, varargin)

Compute the hough transform of a grayscale image without converting to a binary image. (Peng 2006).

img – the input image
varargin
   mskimage – a mask of the areas of interest in the image
varargout
   accumulator of the hough transform
   theta_axis for the accumulator
rho_axis for the accumulator

**Hough Rectangle**

`function [fVect, limits] = RectDetectEdge(srcImage, X, Y, Width, Height, theta_lim, rectwidth, rectheight, tolerance)`

Implementation of the rectangular Hough transform. This works by finding pairs of lines that have the same angle +/- theta_lim. These peaks are matched with other sets of peaks that are perpendicular to the original two peaks.

- srcImage - input image
- X,Y - X,Y coordinates of top of search area
- Width - Width of search area
- Height - Height of search area
- theta_lim - tolerance on theta values (+/- theta_lim degrees)
- rectwidth - width the rectangle must have
- rectheight - height the rectangle must have
- tolerance - the tolerance on rectwidth and rectheight (+/- pixels)

- fVect – X,Y coordinates of the rectangle, angle, width, height, and strength
- limits – numerical limits on fVect

**Rectangle (Generalized Hough Transform Method)**

`function [fVect, limits] = GHT_Rectangle_Detect(srcImage, X, Y, Width, Height, rect_width, rect_height)`

This is a method to implement the GHT Rectangular Detector see Davies Machine Vision: Theory Algorithms Practicalities for details.

- srcImage – input image
- X,Y – the X,Y coordinates of the area of interest on the input image
- Height, Width – the width and height of the area of interest on the input image
- rect_width – the width of the rectangle being searched for
rect_height – the height of the rectangle being searched

fVect – X,Y location of the rectangle, theta angle of the rectangle and strength of the rectangle
limits – numerical limits on the fVect values

**Corner**

\[ \text{function } [fVect, \text{limits}] = \text{Detect\_Corner\_Median(srcImage, X, Y, Width, Height)} \]

This is a function to perform corner detection on an image. It works by subtracting the original image from its median see Davies (2005) for a reference on this technique.

srcimage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image

fVect – X,Y coordinates of the strongest corner, value of corner, 0
limits – numerical limits on the fVect values

**Hole**

\[ \text{function } [fVect, \text{limits}] = \text{HoughHoleDetect(srcImage, X, Y, Width, Height, Rfind, Polarity, HoG)} \]

A hough hole detection algorithm. Sobel edge detection is applied and then a hough transform is used to find the hole.

srcimage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
Rfind – the radius of the hole to find
Polarity – not used
HoG – whether to apply Histogram of Gradients prefiltering

fVect – X,Y coordinates of centre, radius, strength
limits – numerical limits on the fVect values

\[ \text{function} \ [\text{Accumulator}, \ xdim, \ ydim] = \text{houghcircle2}(\text{Imbinary}, r) \]

Fast implementation of Circular Hough-Transform (pFragner 2006).

Imbinary – binary input image
r – the radius of the hole

Accumulator – the Hough accumulator
xdim, ydim – the x and y dimensions of the accumulator

\textbf{Circle}

\[ \text{function} \ [fVect, \ limits] = \text{HoughCircleDetectGrd}(\text{srcImage}, X, Y, \text{Width}, \text{Height}, \text{Rin}, \text{Rout}, \text{HoG}) \]

Wrapper for the Hough gradient technique.

srcImage – the input image,
X, Y – the X, Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
Rin, Rout – the inner and out radius of the circle being searched for
HoG – whether to use Histogram of Gradients prefiltering

fVect – X, Y coordinates of the centre, radius strength
limits – numerical limits on the fVect values

\[ \text{function} \ [\text{varargout}] = \text{CircularHough_Grd}(\text{img}, \text{radrange}, \text{varargin}) \]

Detect circular shapes in a greyscale image. Resolve their center positions and radii (Peng 2006).

img – input image

167
radrange – the range of possible radii

varargout
  circle centre
  radius
  strength

SemiCircle

function [fVect, limits] = HoughSemiCircleDetectGrd(srcImage, X, Y, Width, Height, Rin, Rout, Polarity, HoG)

Wrapper for the Hough semi circle gradient routine

srcimage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
Rin, Rout – the inner and out radius of the circle being searched for
Polarity – whether a top semi circle or bottom semi circle is being found
HoG – whether to use Histogram of Gradients prefiltering

fVect – X,Y coordinates of the centre, radius strength
limits – numerical limits on the fVect values

function [varargout] = SemiCircularHough_Grd_Mod(img, radrange, polarity, varargin)

Detect circular shapes in a grayscale image. Resolve their center positions and radii.

img – input image
radrange – the range of possible radii
Polarity – whether a top semi circle or bottom semi circle is being found

varargout
  circle centre
radius
strength

Radial Hole Statistics

\[ \text{function } f\text{Vect, limits} = \text{RadialHoleStats}(\text{srcImage, X, Y, Width, Height, Rfind, RingWidth}) \]

Wrapper function for radialHole (returns radial hole statistics).

\text{srcimage} – the input image,
\text{X, Y} – the X, Y coordinates of the area of interest on the input image
\text{Height, Width} – the width and height of the area of interest on the input image
\text{Rfind} – the radius of the circle
\text{ringwidth} – the width of the rings

\text{fVect} – inner, middle, outer hole statistics, change between inner and outer
\text{limits} – numerical limits on the fVect values

\[ \text{function } \text{in\_av mid\_av out\_av change} = \text{radialHole}(\text{img,x,y,radius,ringwidth}) \]

This is a function that will find the radial hole statistics of an image given a centre of a hole.

\text{image} – the input image,
\text{X, Y} – the X, Y centre of the hole
\text{radius} – the radius of the circle
\text{ringwidth} – the width of the rings

\text{in\_av} – the inner average
\text{mid\_av} – the middle ring average
\text{out\_av} – the outer ring average
\text{change} – the difference between \text{in\_av} and \text{out\_av}
**Colour**

\[ fVect, \text{limits} = \text{ColorBlob}(\text{srcImage}, X, Y, \text{Width}, \text{Height}, \text{Resolution}, \text{ColorID}) \]

Extracts largest region containing one hue.

srcimage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
Resolution – number of hues to look for
ColorID – the specific hue to look for

fVect – X,Y centre of colour blob, hue, number of pixels
limits – numerical limits on the fVect values

\[ \text{function \{colorMatrix\} = segHSV(\text{srcimage}, \text{numSect}, \text{colorIdx}, \text{numRtn}, \text{satThres}, \text{minSize})} \]

Divides the source image into HSV and then segments into NUMSEG segments. Blobs are extracted form the segments corresponding to the entries in the vector COLORIDX and returned in order of area up to a total of NUMRTN entries.

srcImage – input image
numSect – number of hues to look for
colorIdx – target hue
numRtn – the number of hues to return
satThres - the minimum saturation a colour must have to be considered a colour
minSize - the minimum colour patch size to be processed

colorMatrix – output matrix of colour values
### Colour Stats

*function [fVect, limits] = ColourStats(srcImage, X, Y, Width, Height)*

Wrapper function for average_RGB.

*srcimage – the input image,*

*X,Y – the X,Y coordinates of the area of interest on the input image*

*Height, Width – the width and height of the area of interest on the input image*

*fVect – average red value, average blue value, average green value, average greyscale value*

*limits – numerical limits on the fVect values*

*function [stats] = average_rgb(srcimage)*

Function to produce useful statistics on the colours used in an image. This function will calculate various average values for the image. These are: average red, green and blue components and average lightness from the greyscale image.

*srcimage - the input image*

*rav - average red value in the image*

*gav - average green value in the image*

*bav - average blue value in the image*

*bwav - average lightness value (from greyscale image)*

### PCA Colour

*function [fVect limits] = PCAColour(srcImage, X, Y, Width, Height)*

A function to apply PCA to provide better differentiation of the colour channels in a colour image.

*srcimage – the input image,*

*X,Y – the X,Y coordinates of the area of interest on the input image*

*Height, Width – the width and height of the area of interest on the input image*
fVect – eigenvalue 1, eigenvalue 2, eigenvalue 3, 0
limits – numerical limits on the fVect values

**Centroid**

function \([fVect \ limits] = PreProcessCentroid(srcImage, X, Y, Width, Height, method)\)

This is a wrapper including preprocessing for the centroid function.

srcimage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
method – method of centroid prefiltering

fVect – X,Y location of centroid
limits – numerical limits on the fVect values

function \([xcoord,ycoord] = centroid(inimage)\)

This is a function to determine the centroid of an image.

inimage – input image

xcoord, ycoord – the x and y coordinates of the centroid.
C.2.2 Regions of Interest

ROI – Basic

function [fVect, ROI_Img] = ROI_Extract_BASIC(srcImage, X, Y, Width, Height)

Region of Interest Based Extraction method that will return the pixels in the region of interest in a
matrix after converting them to greyscale.

srcImage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
fVect - [x1, x2, y1, y2] coords of extracted region
ROI_Img – the region of interest returned

ROI – Scale

function [fVect, ROI_Img] = ROI_Extract_Scale(srcImage, X, Y, Width, Height, ScaleFactor)

Region of Interest Based Extraction method that will scale the region of interest down by a given
scale factor, to reduce the number of inputs that need to be processed. Outputs a greyscale image.

srcImage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
ScaleFactor – the amount by which the image will be down scaled.

fVect - [x1, x2, y1, y2] coords of extracted region
ROI_Img – the region of interest returned
ROI – Edge

function [fVec, ROI_Img] = ROI_Extract_Edge(srcImage, X, Y, Width, Height, ScaleFactor)

Region of Interest Based Extraction method that will scale the region of interest down by a given scale factor, to reduce the number of inputs that need to be processed. It will also perform edge detection on this output. Outputs a greyscale image.

srcImage – the input image,
X,Y – the X,Y coordinates of the area of interest on the input image
Height, Width – the width and height of the area of interest on the input image
ScaleFactor – the amount by which the image will be down scaled.

fVec - [x1, x2, y1, y2] coordinates of extracted region
ROI_Img – the region of interest returned

C.3 Classifier Algorithms

C.3.1 ANFIS

The subroutine used to train the ANFIS network is called Train_ANFIS it is a wrapper function for the anfis routine built into MATLAB®.

function [FIS4, min_chk_err] = Train_ANFIS(inTraining, outTraining, Training, Checking, curclip)

helper function for ANFIS training

inTraining – input training set
outTraining – training targets
Training – the training portion of the data set
Checking – the checking portion of the data set.

FIS4 – output trained FIS structure
min_chk_err – the minimum RMS error found for the output trained FIS structure based on the checking data set.

The following is code snippet that details an outline of the training routine. Note genfis2 & anfis are MATLAB® subroutines.

```matlab
FIS1 = genfis2(inTraining, outTraining, 0.5); %generate the fis structure
FIS1.andMethod = 'min'; % Modify FIS style for easier retraining
FIS2 = redSubFis(FIS1, 0.5); %eliminate overlapping clusters
% train the anfis system
[FISTemp, ERR, STEPS, FIS4, CHKERR] = anfis(Training, FIS2, [nEpoch,0,NaN,NaN,NaN],[0,0,0,0], Checking);
min_chk_err = min(CHKERR); %output the error
```

```matlab
function [smallFis] = redSubFis(bigFis, mergeThres);

Takes a FIS structure generated by subtractive clustering (genfis2) and reduces the number of membership functions and rules by merging membership functions with similar ranges and adjusting rules accordingly. Rules which become identical after this process are merged.

mergeThres - determines the required score of a member at the centre point of the current member for merging to occur.
bigFis – the input FIS
smallFIS – the output FIS
```

**C.3.2 Neural Network**

The following code performs the neural network training. Both newff and train are built in MATLAB® functions in the Neural Network toolbox. ins is the input data set. targets is the targets outputs for the input dataset. 20 is the number of nodes in the hidden layer.
nnfeatstruct = newff(ins,targets,20);
nnfeatstruct = train(nnfeatstruct,ins,targets);

C.3.3 PCA ANFIS

For the PCA ANFIS technique first PCA is applied and the system is trained using \textit{Train\_ANFIS} (see C.3.1). The following code applies PCA. Note \texttt{ins} is the input data.

\begin{verbatim}
[ins_rr,RemovedRows] = removeconstantrows(ins); %remove constant rows
[ins_norm,ps1] = mapstd(ins_rr); % normalize the data
[intrans,ps2] = processpca(ins_norm,0.05); %PCA process the data
\end{verbatim}

\textit{removeconstantrows} removes any constant rows (all values are the same). \textit{mapstd} normalizes the data against itself to prepare the data for applying PCA. \textit{processpca} applies PCA to the input data. All three functions are MATLAB® function from the Neural Network toolbox.

C.3.4 PCA Neural Network

The PCA NN technique operates the same as the PCA ANFIS (see C.3.3) technique except a Neural Network is trained. The following code performs this process. See C.3.2 and C.3.3 for an explanation on how this code works.

\begin{verbatim}
[ins_rr,RemovedRows] = removeconstantrows(ins); %remove constant rows
[ins_norm,ps1] = mapstd(ins_rr); % normalize the data
[intrans,ps2] = processpca(ins_norm,0.05); %PCA process the data

nnfeatstruct = newff(intrans,targets,20);
nnfeatstruct = train(nnfeatstruct,intrans,targets);
\end{verbatim}
C.3.5 Eigenimage ANFIS

The following code applies the Eigenimage routine, which is a PCA technique. Again \textit{ins} is the input. See Section C.3.3 for an explanation of the PCA code. After this \textit{Train_ANFIS} is called to training the ANFIS system

\begin{verbatim}
[ins_rr,RemovedRows] = removeconstantrows(ins'); % remove constant rows
[ins_norm,ps1] = mapstd(ins_rr); % normalize the data
[intrans,ps2] = processpca(ins_norm',0.01); % PCA process the data

% multiply the input images by the eigenimages to get
% coefficients in the eigenspace as inputs for the neural
% network

eigenins = intrans*ins_norm;
\end{verbatim}

C.3.6 Eigenimage Neural Network

This following code shows how the Eigenimage Neural Network technique is applied. An explanation of the PCA / Eigenimage code is found in Sections C.3.3 and C.3.5. An explanation of the Neural Network code is found in C.3.2.

\begin{verbatim}
[ins_rr,RemovedRows] = removeconstantrows(ins'); % remove constant rows
[ins_norm,ps1] = mapstd(ins_rr); % normalize the data
[intrans,ps2] = processpca(ins_norm',0.01); % PCA process the data

% multiply the input images by the eigenimages to get
% coefficients in the eigenspace as inputs for the neural
% network

eigenins = intrans*ins_norm;
nnfeatstruct = newff(eigenins,targets',20);
nnfeatstruct = train(nnfeatstruct,eigenins,targets');
\end{verbatim}
The following is a listing of the feature selection related subroutines. Section 4.4 contains more details on these methods.

**C.4.1 Exhaustive Method**

```matlab
function [bestFeat, globalMin, bestLabels, stopfeatnum, stopfeatpos] = optimizeDims(inFis, trainMatrix, checkMatrix, threshold, labels, curclip, startfeatnum, startfeatpos, startGlobalMin)
```

Takes a FIS structure (INFIS) and a training set (INMATRIX/OUTMATRIX) and attempts to find the combination of input dimensions that gives the lowest total error.

optional inputs - `startfeatnum` and `startfeatpos` are used to index where the optimization left off

For now, the following is assumed:
- minimum error considered after a constant number of epochs
- exhaustive search is used (no genetic or probabilistic)
- stops once error is less than threshold

- `inFIS` – the input FIS structure
- `trainMatrix` – the training Matrix
- `checkMatrix` – the checking Matrix
- `threshold` – the error threshold
- `labels` – labels for the input features
- `curclip` – the current clip
- `startfeatnum` – optional parameter for what number of features to resume the optimisation
- `startfeatpos` – optional parameter for what pos in the given number of features to resume
- `startGlobalMin` – optional parameter for the global error to resume the optimisations at

- `bestFeat` – the best set of features
- `globalMin` – the error associated with the best feature set
- `bestLabels` – labels of the features in the best Feature set
C.4.2 Forward Method

```matlab
function [bestFeat, globalMin, bestLabels] = optimizeForward(inFis, trainMatrix, checkMatrix, 
threshold, labels, curclip)
```

Takes a FIS structure (INFIS) and a training set (INMATRIX/OUTMATRIX) and attempts to find the combination of input dimensions that gives the lowest total error. For now, the following is assumed:

- minimum error considered after a constant number of epochs
- forward search is used (add features from no feature set)
- stops once error is less than threshold or all possibilities tried

inFIS – the input FIS structure
trainMatrix – the training Matrix
checkMatrix – the checking Matrix
threshold – the error threshold
labels – labels for the input features
curclip – the current clip

bestFeat – the best set of features
globalMin – the error associated with the best feature set
bestLabels – labels of the features in the best Feature set

C.4.3 Forward (many) Method

```matlab
function [bestFeat, globalMin, bestLabels] = optimizeForwardN(inFis, trainMatrix, 
checkMatrix, threshold, labels, numResultsHeld, curclip)
```

Takes a FIS structure (INFIS) and a training set (INMATRIX/OUTMATRIX) and attempts to find the combination of input dimensions that gives the lowest total error. For now, the following is assumed:
- minimum error considered after a constant number of epochs
- forward search is used (add features from no feature set)
- stops once error is less than threshold or all possibilities tried

inFIS – the input FIS structure
trainMatrix – the training Matrix
checkMatrix – the checking Matrix
threshold – the error threshold
labels – labels for the input features
numResultsHeld – the number of results to check
curclip – the current clip

bestFeat – the best set of features
globalMin – the error associated with the best feature set
bestLabels – labels of the features in the best Feature set

C.4.4 Exhaustive with Prefiltering Method

function [perms] = bestPermOpt(curclip,numfeats)

This method takes the best numfeats and calculates all the permutations of them. The best features are found based on the rms of 1 features. It calls calc_permute

curclip - clip to perform feature search on
numfeats – the number of features to select in prefiltering.

perms – the output of the feature selection

function perms = calc_permute(features,curclip)

This will calculate all the permutations of a given optimisation and store it. This needs to be done while running an opened QVision file.
features – the features to search exhaustively
curclip – the clip to perform the feature selection on
perms – the output of the feature selection

C.5 Performance Evaluation

C.5.1 RMS

The RMS error was calculated using the RMS equation detailed in 3.4.2.

C.5.2 ROC

To plot the ROC curves the MATLAB® function \textit{plotROCCurve} was called. More details on ROC curves are presented in section 2.6.