

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

A novel approach to recognition of the detected moving
objects in non-stationary background using heuristics
and colour measurements

A thesis presented in partial fulfilment of the requirement for the degree
of

Master of Engineering

At

Massey University,
Albany, New Zealand

Kartikay Lal

2017

Abstract

Computer vision has become a growing area of research which involves two fundamental steps, object detection and object recognition. These two steps have been implemented in real world scenarios such as video surveillance systems, traffic cameras for counting cars, or more explicit detection such as detecting faces and recognizing facial expressions. Humans have a vision system that provides sophisticated ways to detect and recognize objects. Colour detection, depth of view and our past experience helps us determine the class of objects with respect to object's size, shape and the context of the environment. Detection of moving objects on a non-stationary background and recognizing the class of these detected objects, are tasks that have been approached in many different ways. However, the accuracy and efficiency of current methods for object detection are still quite low, due to high computation time and memory intensive approaches. Similarly, object recognition has been approached in many ways but lacks the perceptive methodology to recognise objects.

This thesis presents an improved algorithm for detection of moving objects on a non-stationary background. It also proposes a new method for object recognition. Detection of moving objects is initiated by detecting SURF features to identify unique keypoints in the first frame. These keypoints are then searched through individually in another frame using cross correlation, resulting in a process called optical flow. Rejection of outliers is performed by using keypoints to compute global shift of pixels due to camera motion, which helps isolate the points that belong to the moving objects. These points are grouped into clusters using the proposed improved clustering algorithm. The clustering function is capable of adapting to the search radius around a feature point by taking the average Euclidean distance between all the feature points into account. The detected object is then processed through colour measurement and heuristics. Heuristics provide context of the surroundings to recognize the class of the object based upon the object's size, shape and the environment it is in. This gives object recognition a perceptive approach.

Results from the proposed method have shown successful detection of moving objects in various scenes with dynamic backgrounds achieving an efficiency for object detection of over 95% for both indoor and outdoor scenes. The average processing time was computed to be around 16.5 seconds which includes the time taken to detect objects, as well as recognize them. On the other hand, Heuristic and colour based object recognition methodology achieved an efficiency of over 97%.

Acknowledgements

First and foremost, I would like to thank my supervisor Dr. Khalid Arif for his support and valuable insights throughout the duration of the thesis.

I would also like to thank my family for their support and understanding.

Contents

Abstract	i
Acknowledgements	ii
Chapter 1 Introduction	1
1.1 Background	1
1.2 Problem Statement	2
1.3 Proposed solution and novelty	3
1.4 Outline of Thesis	3
Chapter 2 Literature Review	4
2.1 Image Registration, Optical Flow computation and clustering	5
2.2 Object Recognition	12
2.3 Summary of findings	21
Chapter 3 Proposed Method	24
3.1 Auto Image contrast	27
3.2 Evaluation of Edge detection algorithms	30
3.3 Evaluation of Feature detection techniques and K-means clustering algorithm	35
3.4 Computing Camera-Shift	44
3.5 Optical Flow computation	48
3.6 Elimination of outliers	55
3.7 Clustering	60
3.8 Colour measurements	66
3.9 Heuristics based Artificial Intelligence	73
3.9.1 TensorFlow and heuristics approach to object recognition.....	76
Chapter 4 Testing Results	78
4.1 Object detection using Optical flow and Clustering	78
4.1.1 Results obtained by researchers for optical flow and object detection	78

4.1.2	Results for object detection using the proposed algorithm	81
4.2	Object recognition based on Heuristics and Colour measurements	101
4.3	Scene recognition using TensorFlow	117
4.4	Summary of Results & comparison with literature	124
Chapter 5	Conclusion and future work.....	126
References	129

List of Figures

<i>Figure 2-1: Simplified Conceptual Diagram of moving object detection on non-stationary background.....</i>	<i>4</i>
<i>Figure 2-2: Results from [19] (top) and [20] (bottom) showing background subtraction and deleting background features respectively.....</i>	<i>7</i>
<i>Figure 2-3: Results from [22] and [24] showing optical flow using pyramidal LK method.....</i>	<i>8</i>
<i>Figure 2-4: Two consecutive frames with objects detected that were used by [30].....</i>	<i>11</i>
<i>Figure 2-5: Results from [40] (left column) and [41] (right column).....</i>	<i>14</i>
<i>Figure 2-6: Results extracted from [42]</i>	<i>15</i>
<i>Figure 2-7: Results extracted from [44]</i>	<i>16</i>
<i>Figure 2-8: Original image, image contours, super features identified, taken from [47].....</i>	<i>17</i>
<i>Figure 2-9: Results extracted from [49] (top) and [50] (bottom)</i>	<i>19</i>
<i>Figure 3-1: Algorithm Design.....</i>	<i>25</i>
<i>Figure 3-2: Shows original image (a) and its histogram equalized counterpart image (b). Plotted histogram (a) and histogram (b) for images (a) and (b) respectively. Plotted SURF feature points for the images (a) and (b)</i>	<i>29</i>
<i>Figure 3-3: SURF Feature points plotted without edge detection (Top) and feature points detected using each edge detector for Car frame (middle and bottom rows).....</i>	<i>32</i>
<i>Figure 3-4: SURF Feature points plotted without edge detection (Top) and feature points detected using each edge detector for Shooting frame</i>	<i>34</i>
<i>Figure 3-5: Two grayscale frames named Test frame 1 and Test frame 2. The camera is stationary with only the objects (cars) moving.....</i>	<i>35</i>
<i>Figure 3-6: Corners detected using Harris corner detector (top row) with clustering through k-means (bottom row).....</i>	<i>36</i>
<i>Figure 3-7: SURF feature points plotted (top row) with clustering through K-means (bottom row) ..</i>	<i>37</i>
<i>Figure 3-8: Features detected using SIFT feature detector (top row) and clustering through k-means algorithm (bottom)</i>	<i>38</i>
<i>Figure 3-9: Second set of test frames</i>	<i>39</i>
<i>Figure 3-10: Harris, SURF and SIFT feature points with colour coded clusters and their centroids shown from top, middle and bottom respectively</i>	<i>40</i>
<i>Figure 3-11: Harris (top), SURF (middle) and SIFT (bottom) features detected for blurry frames</i>	<i>42</i>
<i>Figure 3-12: Result of running Code snippet 3-3</i>	<i>46</i>
<i>Figure 3-13: Two set of frames displaying matched feature points.</i>	<i>47</i>
<i>Figure 3-14: Optical flow diagram.....</i>	<i>50</i>
<i>Figure 3-15: The matching process of feature points between two consecutive frames using cross correlation.....</i>	<i>51</i>
<i>Figure 3-16: Result of detecting SURF features on images shown in Figure 3-15.....</i>	<i>51</i>
<i>Figure 3-17: Surf plot of x,y peaks as a result of normalized cross-correlation.....</i>	<i>53</i>
<i>Figure 3-18: Optical flow between two pairs of frames named Target (left) and Soccer (right).....</i>	<i>54</i>

Figure 3-19: Showing pair of Target (top) and Soccer (bottom) frames depicting elimination of points using RANSAC	56
Figure 3-20: Motion vectors after elimination of outliers for Soccer frame (top) and Target frame (bottom).....	59
Figure 3-21: Flowchart of the clustering function.....	60
Figure 3-22: Shows two cameramen with feature points plotted in red and cluster centroid marked in yellow	61
Figure 3-23: Shows bounding boxes around each cluster	62
Figure 3-24: Single cluster comprising of all the small clusters.....	62
Figure 3-25: Shows the searching of points in vicinity	62
Figure 3-26: A point marked yellow with <i>intraSearchRadius</i> (yellow) and <i>centroid_disp_limit</i> (orange).....	63
Figure 3-27: Target (top) and Soccer (bottom) frames showing motion vectors after the elimination phase	66
Figure 3-28: Original Soccer frame	67
Figure 3-29: Detected colour regions using HSV thresholding from left to right, red, green, blue, white, gray.....	67
Figure 3-30: Extracted R,G,B and H,S,V planes on first and second rows respectively for the original RGB frame shown in Figure 3-28	68
Figure 3-31: Shows the masking process using the AND operation	71
Figure 3-32: Flowchart of colour detector and colour threshold values in HSV space	72
Figure 3-33: Perceptive approach using object shape, colours in the image and heuristics.....	75
Figure 3-34: Inception model	77
Figure 4-1: Result from [30] showing detection of objects on smooth background	78
Figure 4-2: Results from [25] showing feature extraction (left) and the detected object in the foreground.....	79
Figure 4-3: Results from [20] displays background modelling (top row), object moving in the foreground (middle row), detected object (bottom row)	79
Figure 4-4: Results from [19] depicting detection of moving object while the background is also moving.....	80
Figure 4-5: Result obtained from [22] displaying corner points detected using Harris corner detector (left) and the detected objects (right)	80
Figure 4-6: Results obtained from [24] Optical flow computed using pyramidal LK method.....	81
Figure 4-7: Bike frames, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm (c) moving object detected as a result of optical flow (d) detected object	82
Figure 4-8: Bike frames, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object.....	83

Figure 4-9: Car sequence 1, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	84
Figure 4-10: Car sequence 1, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	85
Figure 4-11: Car sequence 2, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	86
Figure 4-12: Car sequence 2, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	87
Figure 4-13: Aerial sequence 3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	88
Figure 4-14: Aerial sequence 3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	89
Figure 4-15: Car sequence 3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	90
Figure 4-16: Car sequence 3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	91
Figure 4-17: Target sequence, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	92
Figure 4-18: Target sequence, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	93
Figure 4-19: Car frames, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	94
Figure 4-20: Shooting sequence, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) small blocks detected, (d) merged blocks for detected object	95
Figure 4-21: Soccer frames, (a) final motion vectors as a result of elimination of outliers, (b) clustering of points using the proposed clustering algorithm, (c) detected objects	96

Figure 4-22: Indoor frame SM1, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	97
Figure 4-23: Indoor frame SM1, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	98
Figure 4-24: Indoor frame SM3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	99
Figure 4-25: Indoor frame SM3, (a) motion vectors as a result of optical flow computation, (b) clustering of points using the proposed clustering algorithm, (c) moving object detected as a result of optical flow, (d) detected object	100
Figure 4-26: Different colour areas for Lumix frame along with the recognized object in frame (c) shown in purple (top marker).....	103
Figure 4-27: Different colour areas for Target frame along with the recognized object in frame (c) shown in purple (top marker).....	104
Figure 4-28: Different colour areas for Shooting frame along with the recognized object in frame (c) shown in purple (top marker).....	105
Figure 4-29: Different colour areas for Soccer frame along with the recognized objects in frame (c) shown in purple (top marker).....	106
Figure 4-30: Different colour areas for Car sequence 3 frame with the recognized objects in frame (c) shown in purple (top marker).....	107
Figure 4-31: Different colour areas for Car sequence 1 frame with the recognized object in frame (c) shown in purple (top marker).....	108
Figure 4-32: Different colour areas for Car sequence 2 frame with the recognized object in frame (c) shown in purple (top marker).....	109
Figure 4-33: Different colour areas for Aerial sequence frame with the recognized object in frame (c) shown in purple (top marker).....	110
Figure 4-34: Different colour areas for SM1 frame with the recognized object in frame (b) shown in purple (top marker).....	111
Figure 4-35: Different colour areas for SM3 frame with the recognized object in frame (c) shown in purple (top marker).....	112
Figure 4-36: Erroneous heuristic implementation for SM1 and SM3 sequence frames	113
Figure 4-37: Heuristics for test images 1 and 2.....	114
Figure 4-38: Heuristics for test image 3	115
Figure 4-39: Heuristics for test images 4 and 5.....	116
Figure 4-40: Bike frame (top) and Aerial sequence (bottom), showing score for each attribute found in the frames using TensorFlow	118
Figure 4-41: Car frame (top) and SM1 sequence (bottom), showing score for each attribute found in the frames using TensorFlow	119

<i>Figure 4-42 Test image 1 frame (top) and Test image 2 (bottom), showing score for each attribute found in the images using TensorFlow.....</i>	<i>120</i>
<i>Figure 4-43: Test image 3 frame (top) and Test image 4 (bottom), showing score for each attribute found in the images using TensorFlow.....</i>	<i>121</i>
<i>Figure 4-44: Test image 5 (top) and Test image 6 (bottom), showing score for each attribute found in the images using TensorFlow.....</i>	<i>122</i>
<i>Figure 5-1: Conceptual diagram highlighting the new and improved areas</i>	<i>127</i>

List of Tables

<i>Table 2-1: Summary of papers</i>	22
<i>Table 3-1: Number of feature points detected using different edge detection methods</i>	30
<i>Table 3-2: Computational time for each detector</i>	39
<i>Table 3-3: Processing time taken for different optical flow methods</i>	49
<i>Table 3-4: RGB and HSV low/high threshold values</i>	67
<i>Table 4-1: Results of object detection and recognition for various test frames</i>	102

List of Code Snippets

<i>Code snippet 3-1: Matlab code showing automatic equalization for pixel intensity</i>	28
<i>Code snippet 3-2: Extracts SURF feature points and detects edges using Prewitt</i>	33
<i>Code snippet 3-3: Matlab code to compute camera shift</i>	45
<i>Code snippet 3-4: Padding the image for full search block matching with specified padding side</i>	51
<i>Code snippet 3-5: Extraction of (x,y) location and rounding to whole numbers</i>	52
<i>Code snippet 3-6: Acquiring of a small patch of a certain size from the padded image B</i>	52
<i>Code snippet 3-7: Computation of distance between two matched points in image 1 and image 2</i>	52
<i>Code snippet 3-8: Code to perform cross-correlation</i>	53
<i>Code snippet 3-9: Computes the distance between the detected points and the new points</i>	54
<i>Code snippet 3-10: Plotting the vector arrows using the quiver() function</i>	54
<i>Code snippet 3-11: Elimination of vector arrows</i>	57
<i>Code snippet 3-12: Discard vector arrows that are 5 times the length of the common vector size.</i>	58
<i>Code snippet 3-13: Discard all error points that have a value of 0</i>	58
<i>Code snippet 3-14: Function prototype for the clustering function</i>	60
<i>Code snippet 3-15: searching of points in the vicinity of x1,y1, following Figure 3-26. The variables x1,y1 are SURF feature points detected on image 1</i>	63
<i>Code snippet 3-16: Check if x2,y2 lies within intraSearchRadius of x1,y1</i>	64
<i>Code snippet 3-17: Storing centroid locations</i>	64
<i>Code snippet 3-18: Nested for-loops to search for points in vicinity using intraSearchRadius</i>	65
<i>Code snippet 3-19: Function prototype for detection of colour</i>	69
<i>Code snippet 3-20: Converting the images to HSV space using hsvImage() function</i>	69
<i>Code snippet 3-21: Computing mask for each H, S and V plane</i>	69
<i>Code snippet 3-22: Performing AND operation to check where all three are true</i>	70
<i>Code snippet 3-23: SetThresholdsDetectColour() function to set colour thresholds</i>	71
<i>Code snippet 3-24: Detection of colour using thresholds in a switch-case system</i>	71