An Efficient Approach To Object Recognition For Mobile Robots.

AHMED, M Shuja, SAATCHI, Reza <http://orcid.org/0000-0002-2266-0187> and CAPARRELLI, Fabio

Published version

AHMED, M Shuja, SAATCHI, Reza and CAPARRELLI, Fabio (2013). An Efficient Approach To Object Recognition For Mobile Robots. In: BENAVENTE-PECES, Cesar and FILIPE, Joaquim, (eds.) Proceedings of the 3rd International Conference on Pervasive Embedded Computing and Communication Systems. Scitepress, 60-65.

AN EFFICIENT APPROACH TO OBJECT RECOGNITION FOR MOBILE ROBOTS

M Shuja Ahmed, Reza Saatchi and Fabio Caparrelli

Materials and Engineering Research Institute, Sheffield Hallam University, Sheffield, United Kingdom
m_shujaz@hotmail.com, r.saatchi@shu.ac.uk, f.caparrelli@shu.ac.uk

Keywords: Object Recognition, SURF, Harris Features, Multi-Robotics

Abstract: In robotics, the object recognition approaches developed so far have proved very valuable, but their high memory and processing requirements make them suitable only for robots with high processing capability or for offline processing. When it comes to small size robots, these approaches are not effective and lightweight vision processing is adopted which causes a big drop in recognition performance. In this research, a computationally expensive, but efficient appearance-based object recognition approach is considered and tested on a small robotic platform which has limited memory and processing resources. Rather than processing the high resolution images, all the times, to perform recognition, a novel idea of switching between high and low resolutions, based on the "distance to object" is adopted. It is also shown that much of the computation time can be saved by identifying the irrelevant information in the images and avoid processing them with computationally expensive approaches. This helps to bridge the gap between the computationally expensive approaches and embedded platform with limited processing resources.

1 INTRODUCTION

Object recognition, when it comes to robotics, is considered an essential element since, in most robotic applications, the robots are required to look for and recognize the objects of interest in the environment to achieve the given objectives. The use of vision sensors in robotic applications has provided many solutions for object recognition. Object recognition is considered as one of the most challenging problems in computer vision. It presents several challenges such as, view point changes, intensity variations, occlusions and background clutter. Its use in mobile robotics applications introduces an important challenge given by the constraints for execution time (i.e. computational complexity). In real world scenarios, robots are normally equipped with high powered processing systems which can fulfil the computational demands of the object recognition algorithms. For such robots the choices of object recognition techniques may be diverse and computationally expensive object recognition techniques may be simply adopted with minor changes. But when it comes to small robots with limited processing resources, the task becomes challenging because the algorithm fails to achieve real time performance.

In computer vision, many object recognition approaches have been introduced. The content based approaches (i.e. using colour, texture and shape) and geometric approaches (i.e. using affine or projective transformations) are computationally less expensive, but does not give good performance in varying lighting conditions. On the other hand, context based and appearance based approaches (e.g. using SIFT, SURF or PCA features) are computationally intensive. These are less sensitive to the changes in scale, rotation, skew and lighting conditions. In most of the recent research (Lowe, 2004), appearance-based approaches are adopted. In (Nayerlaan and Goedeme, 2008), an appearance based approach, using SURF (Speeded Up Robust Features) features, have adopted to perform traffic sign recognition in an embedded system environment. In the beginning, the system was given training using SURF features, extracted from the database of traffic sign images. In the database, the number of images were large which resulted in too many features. This reduced the speed of feature matching process in the recognition stage. To reduce the number of features, the feature space clustering approach (Tuytelaars et al., 2008) can be utilised. Feature space clustering is also called feature space quantization in some approaches (Asanza and Wirnitzer, 2010) and it provides significant saving in memory usage which makes the approach suitable for the implementation on embedded systems. Some researchers have used probabilistic models on the top of SURF feature based approaches to perform robot localisation and mapping (Cummins and Newman, 2010). In comparison to this, in some approaches (Chrysanthakopoulos and Shani, 2010), the authors relied on Harris features to reduce the computational time and used Partially Observable Markov Decision Process (POMDP) probabilistic methods to track the probability distribution of the robot's where-about. In another work (Ramos et al., 2010), SURF features were further processed using Conditional Random Fields (CRF) algorithm to discard those features which do not show geometric consistency. In (Ricardo and Pellegrino, 2010), a detailed comparison of SIFT, PCA-SIFT and SURF feature based approaches is presented and SURF was found to be the fastest to compute features. Authors in (Juan and Gwun, 2009) used K-Nearest Neighbour (KNN) approach for feature space clustering and Random Sample Consensus (RANSAC) to get rid of outliers and showed that the selection of methods to perform recognition mainly depends on the target application. In (Krose et al., 2001), linear image features extracted using Principal Component Analysis (PCA) were used for object recognition. In PCA, the eigenvectors of the image are computed and are used as an orthogonal basis for individual image representation. The motivation of using eigenvectors is that only few of them are required for recognition, but this approach is very sensitive to the effect of perceptual aliasing. In another work, a fast gradient based feature extraction approach was used (Dillmann et al., 2007) together with PCA, to perform the recognition task. Similar to the work done by (Nayerlaan and Goedeme, 2008) and (Asanza and Wirnitzer, 2010), the feature space was quantized using a k-means algorithm. The system used for recognition was a 3GHz high powered processing system which takes 350 milli-seconds on an average to recognise the object. This seems a slow performance considering the power of the processing system. Another similar approach using SIFT feature based recognition is presented by (Siegwart et al., 2010) where features were dynamically assigned different weights, according to the uncertainty associated with them and finally avoid using those features which does not show good recognition results. It is to be noted that, in most of the research where appearance based object recognition is employed, a high performance system is used.

This study suggests some techniques following which, the advantages of computationally expensive algorithms can be enjoyed in small robotic systems to achieve object recognition. This study presents an optimised implementation of SURF features based recognition approach which can show acceptable performance on small robotic platforms without sacrificing the recognition performance. We have used a group of small robots and show that how multiple robots look for 3D and 2D objects of interest in an unknown controlled environment. For recognition purposes, all robots will be given a common visual vocabulary. To help achieve this task efficiently, all robots will be sharing their knowledge with each other. Through wireless communication medium, every robot will be updating other team members about the number of objects found. This way, other team members will not search for the objects found by another robot.

2 METHODOLOGY

The aim of this study was to use the advantages of computationally expensive appearance based recognition approaches and make them run efficiently using a group of small robots. Based on the literature review, the SURF features based approach was found to be the fastest to compute and favourable for implementation on an embedded system. There are a number of open source implementations of SURF feature extraction and matching algorithms. OpenCV (Kaehler and Bradski, 2008) is an open source computer vision library which provides a possible implementation of the SURF algorithm. Another implementation is found in OpenSURF (Evans, 2009) library which is a faster and a more optimized implementation of SURF as compared to OpenCV. In this study, we decided to use the OpenSURF library as a reference implementation of SURF algorithm. To perform experiments, a group of SRV robots were used which are developed by Surveyor Corporation Inc. The SRV robots used a 16/32-bit Blackfin BF537E processor as on-board processing unit. For code execution on the processor, an open source customised version of linux operating system, called uClinux (micro controller Linux), was used. The SURF algorithm provided by the OpenSURF library, was cross compiled for the target Blackfin processor using GNU cross compilation tool-chains on a Linux based development platform. At the beginning, when the cross compiled SURF algorithm ran on Blackfin processor, it took 33 seconds to process an image with 320x240 pixels resolution. There were two main reasons for this slow processing speed. One was the expected computationally expensive nature of the algorithm. Second, the lack of Floating Point Unit (FPU) on the Blackfin processor and the extensive use of floating point operation by the SURF algorithm. To reduce the execution time of the SURF algorithm, code optimisation was performed. For further performance improvement, image data elimination and use of multiple resolutions were also made. These are explained in the following sub-sections.

2.1 Optimisation of SURF for the Blackfin Processor

This section discusses the customisation of the SURF algorithm for the target Blackfin processor based embedded system. Processor specific code customisation requires a detailed study about the architecture of the underlying embedded system. By exploring the architectural features of the embedded system, the full processor performance can be achieved through the software. For example, in this study, Blackfin processor is selected, and since this processor is a fixed point processor it is recommended to avoid floating point operation. The use of uClinux operating system on the target Blackfin processor makes the customisation more challenging as it allows 1.31 fixed point operations only. This restriction requires that all the data in the algorithm should be normalised at every point in the program flow in order to guarantee that data values lies within the range of -0.9 to +0.9. If values exceed these limits then unexpected results can occur.

The Blackfin code optimisation can be done in three different phases which are usually followed in the following order. These phases are Compiler optimization, System optimization and Assembly optimization as detailed by (Katz et al., 2005). After performing these optimisation steps, the improvements achieved in the algorithm's execution performance is shown in Table 1. When no optimization was applied, the program took 33sec to process a single image frame. Compiler optimization (i.e. using fast-math and mfast-fp floating point libraries for Blackfin processor) brought the execution time to 12sec per frame. For further reduction in the execution time, the detail timing analysis for the different portions of the program was done. The parts of the program which were taking more time to execute, were identified. These parts of the program were customised by exploiting the fixed point architecture of the Blackfin processor. Therefore, 1.31 fixed point operations were adopted in the place of floating point operations. This helped in reducing the time to 3 sec per frame. The main reason for the significant reduction in time from 12sec to 3sec was the use of 1.31 fixed point operation libraries which are optimised for Blackfin processor and further customised in assembly language. After eliminating the routing of redundant data or by optimizing the data flow in the algorithm, the time was further reduced to 2.8sec per frame. Further reduction in time can be achieved by using the lower resolution image. As lower resolution image provides less details of the object in the image, so SURF algorithm produce less number of features and recognition performance also drops. As shown in Table 1, the processor specific optimisation is an essential step to achieve real time performance as it significantly reduces the execution time of the algorithm. However, further reduction in the execution time is still required.

Table 1: Optimisation of SURF for the Blackfin Processor.

Optimisation Steps	**Execution Time (milli-sec)**
No Optimisation	33000
-fast-math emulation	31100
mfast-fp emulation	12200
1.31 fixed point library	2995
Data flow management	2800
Low Resolution	750

2.2 Image Data Elimination and Resolution Switching

In the second stage, regions in the image are identified which are redundant and does not contribute in the generation of SURF features. For example, smooth image regions and texture less surfaces in the image does not generate any reliable SURF features. So by not processing these image regions, with computationally intensive SURF algorithm, can reduce the image processing load by a large factor. To eliminate these redundant image regions and identify the image pixels which represent the presence of an object in the image, many light weight feature extraction techniques can be applied. The use of simple edge detector (e.g., canny) can also be made to identify the pixels, but they also detects edges caused by straight lines. In this study, we have used Harris feature extraction algorithm. The reason for using Harris algorithm is that, Harris features detects corners in the image which provides more reliable features.

At the beginning, the whole image was divided into two portions, the top and the bottom. The top portion always lies outside the arena and is causing features from the environment which is outside the arena (see Figure 1). Therefore, the top portion was not considered for processing. Another way of avoiding the outside environment was to use higher arena boundaries. The top portion, separated by the thick Blue line, is shown in Figure 1a and 1b. The bottom portion, lying under the thick Blue line, was further divided into three portions that is middle, left and right (middle, left and right portions shown are separated by the thin blue line) and features extraction is performed using the Harris algorithm. Extracted image features are also shown in Figure 1a and 1b. The centroids of the feature points were computed from the middle, left and the right portions and the windowed images were extracted. These windowed images contain the objects in the image. These windowed images are shown by the red boundary lines. Now SURF features are computed for these extracted windowed images. In these windowed images, while extracting SURF features, again only those pixels were considered for processing which were also identified by the Harris feature detection algorithm.

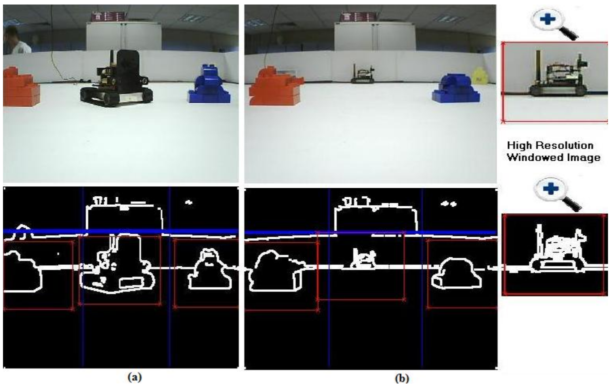

Figure 1: Image data reduction and resolution switching(Ahmed et al., 2012a).

The SURF feature extraction and matching technique works in two stages: training stage and recognition stage. In the training stage, the features of the target object were extracted and kept in memory. In the case of more target objects to be recognized, a database containing the features, resulting from all the target objects, was generated and kept in memory for recognition purposes. In the case of 2D object recognition, only a single object image was required during training stage for extracting SURF features. If the target object was a 3D object, then pose based feature extraction was performed. That is, the images from different poses of the target object were taken and SURF features were computed for all of these images. During the recognition stage, features from all these images were compared with the features extracted from the current view and the best match provides the information about the object and also the direction from which the robot was heading towards the object. This pose based recognition for 3D objects increased the database size by a large factor but keeping the resolution low during the training stage helped in reducing the database size.

To make the recognition technique scale invariant, the SURF algorithm generated the scale-space image pyramid where the input image was iteratively convolved with the Gaussian kernel and, at the same time, the image was sub-sampled iteratively (this reduced the size of the image) (Evans, 2009). During the training stage, if the image resolution was set to 320x240 pixels and the number of times the image was sub-sampled was set to 4, then in the image pyramid, the sequence in which the resolution was down sampled was 320x240, 160x120, 80x60 and, finally, 40x30 pixels. As the target embedded system had a limited memory and processing resources, the training was given in 320x240 pixels resolution so that the resultant features database was smaller in size and can be kept in the memory for recognition purposes. With this resolution, if the object lied a bit far from the robot then it was still recognized. But if it lied further away, then recognition was not possible as it appeared really small in the 320x240 pixels image. To overcome the problem of recognizing the objects lying far from the robot, a multi-resolution analysis was performed. The distance to the objects was measured in 320x240pixels resolution. To measure the distance to the objects, a segmentation based approach was used as addressed in (Ahmed et al., 2012b), where it was originally adopted to perform obstacle avoidance. After measuring the distance, near lying objects were processed in lower resolution and for far lying objects, their positions were determined in the higher resolution image and windowed images were extracted from the higher resolution image. This way, the number of pixels, defining the far lying object, increased and made the recognition possible. In other words, to increase the recognition performance, the objects detected in the image were processed in different resolutions depending upon their distance from the robot. This concept is explained in Figure 1b. The two objects on the left and right side are placed closer to the robot vision system. The windowed image is extracted from the lower resolution image(i.e. 320x240 pixels) for SURF features extraction and matching purposes. The object in the center of the image (i.e. another robot) is lying far from the robot so a higher resolution analysis is performed and windowed image is extracted from the high resolution image, as shown on the right side of Figure 1b.

3 RESULTS

For performance analysis, a test was planned in which the task was to recognise three 2D objects (i.e., two building images and one plain text “Replicator”) and one 3D object (i.e., another robot) as shown in Figure 2. During the training stage, a single image was used to extract SURF features of 2D object, whereas, for 3D object, images from 16 different poses of the robot were used. The objects' features were provided to each robot participating in the experiment. In the current experiment, only two robots were used but the process can also scale-up to larger number of robots as long as each robot is provided the features of the target objects. The two robots were programmed to move randomly in the unknown structured environment and search for the objects of interest collectively. To perform the collective operation, the robots also shared information about the number of objects found and which they were still looking for, over a wireless network. When one robot found the object of interest, it informed the other robot to remove this object from the search list so that redundancy of searching the same object by two robots could be avoided. The placement of the objects of interest in the test arena is shown in Figure 2. Apart from the target objects, some unknown objects can be noticed in the arena as shown in Figure 2. These objects will be treated as obstacles because the features extracted from these objects will not match with the features provided during the training stage.

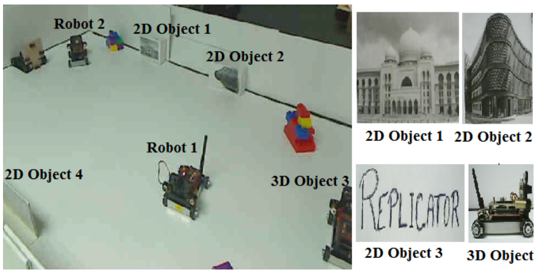

Figure 2: Position of robots and objects of interest before the experiment was performed.

In this experiment, while collectively searching for the objects, robot 1 found the 2D object 1, object 2 and 3D object 3. Robot 2 found 2D object 4. The positions of the robots, when they found the objects, are shown in Figure 3. During this experiment, robot 1 missed the 3D object in the first attempt and successfully found it in the second attempt. The sequence of robot 1 positions when it missed object 3, are shown in Figure 4. In scene image(a), robot 1 detected object 3. Then, it was required to get closer to object 3 to confirm its presence. In scene image(b), robot 1 got closer to object 3. Object 3 was detected on its right side, so in scene image(c), robot 1 turned right. After detecting the object in scene image(c), the robot moved towards the object. But it moved a bit more in the forward direction such that it left object 3 on its left side undetected. Now, in scene image(d), only a small portion of the 3D object was in its field of view and it was not enough for recognition.

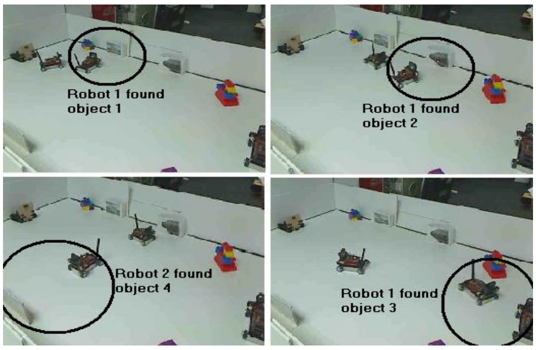

Figure 3: Position of the robots when they recognize the object in the environment.

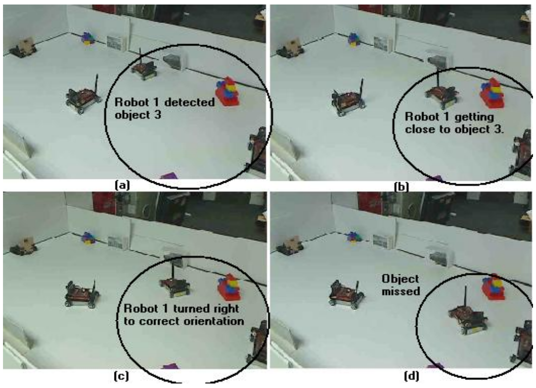
Robot 1 detected object 3
(a)
Robot 1 getting close to object 3.
(b)
Robot 1 turned right to correct orientation
(c)
Object missed
(d)

Figure 4: Sequence of robot 1 positions when it detected and then missed object 3.

Similarly, robot 2 also missed object 1 after detection. This happens because of the interruption from the robot 1. The sequence of robots 1 and 2 positions, when robot 2 missed object 1, are shown in Figure 5. In scene image(a), robot 2 detected object 1 while robot 1 was also nearby. In scene image(b), robot 1 also detected object 1 on its right side. In scene image(c), robot 1 corrected its orientation towards object 1 and robot 2 moved towards object 1. In scene image(d), robot 1 moved towards object 1 but it also moved between robot 2 and object 1. Now in scene image(d), it can be seen that robot 1 partially blocked robot 2 field of view such that robot 2 could not see object 1. Due to this robot 2 missed object 1 and then it was found by robot 1. It is observed that once the robot found the 2D objects, then they hardly missed them while getting closer to the objects. But in the case of 3D object, sometimes the robot detects the object and then missed it while getting closer to it. This happens because the recognition algorithm switches resolution based on the distance to the ob-

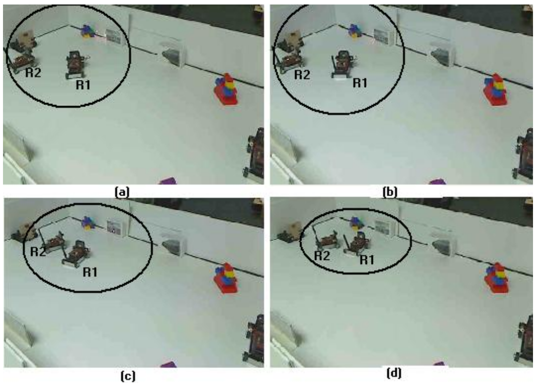

Figure 5: Sequence of robot 2 positions when it detected and then missed object 1.

ject and if at some point, the distance information is wrong then the robot misses the object.

4 CONCLUSION

In this research, an implementation of the computationally expensive object recognition approach on a small robotic platform is addressed. It is concluded that information pre-processing and fully utilizing the architectural features of the target platform can make a big difference in the execution performance of the algorithm. It is shown that, the algorithm performance, to extract and match SURF features, improved from 33 seconds to 750 milli-seconds. Further improvement in the performance can be achieved by coding critical parts of the algorithm in assembly language. It is noticed that, as the size of the visual vocabulary grows, the recognition performance may degrade. To overcome this, clustering of the feature space would be required which will make the algorithm suitable for embedded system implementation.

ACKNOWLEDGEMENTS

Funded by EU-FP7 research project REPLICATOR.

REFERENCES

Ahmed, M., Saatchi, R., and Caparrelli, F. (2012a). Vision based object recognition and localisation by a wireless connected distributed robotic systems. In *Electronic Letters on Computer Vision and Image Analysis Vol. 11, No. 1, Pages 54-67*.

Ahmed, M., Saatchi, R., and Caparrelli, F. (2012b). Vision based obstacle avoidance and odometery for swarms of small size robots. In *Proceedings of 2nd International Conference on Pervasive and Embedded Computing and Communication Systems, Pages 115-122*.

Asanza, D. and Wirnitzer, B. (2010). Improving feature based object recognition in service robotics by disparity map based segmentation. In *International Conference on Intelligent Robots and Systems (IROS). Pages 2716-2720*.

Chrysanthakopoulos, G. and Shani, G. (2010). Augmenting appearance-based localization and navigation using belief update. In *Proceedings of AAMAS 2010, Pages 559-566*.

Cummins, M. and Newman, P. (2010). Fab-map: Appearance-based place recognition and mapping using a learned visual vocabulary model. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), Vol.27 No.6 Pages 647-665*.

Dillmann, R., Welke, K., and Azad, P. (2007). Fast and robust feature-based recognition of multiple objects. In *6th IEEE/RAS International Conference on Humanoid Robots, Pages 264-269*.

Evans, C. (2009). Notes on the opensurf library. In *OpenSurf, Page 25, URL: www.cs.bris.ac.uk/Publications/Papers/2000970.pdf*.

Juan, L. and Gwun, O. (2009). A comparison of sift, pca-sift and surf. In *International Journal of Image Processing (IJIP), Vol.3, No.4, Pages 143-152*.

Kaehler, A. and Bradski, G. (2008.). Computer vision with the opencv library. In *Learning OpenCV, Page 576*.

Katz, D., Lukasiak, T., and Gentile, R. (2005). Enhance processor performance in open-source applications. In *Analog Dialogue, Vol.39*.

Krose, B., Vlassis, N., Bunschoten, R., and Motomura, Y. (2001). A probabilistic model for appearance-based robot localization. In *In First European Symposium on Ambience Intelligence (EUSAI), Pages 264-274*.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision, Vol. 60. No.2 Pages 91-110*.

Nayerlaan, J. and Goedeme, T. (2008). Traffic sign recognition with constellations of visual words. In *International conference on informatics in control, automation and robotics - ICINCO*.

Ramos, F., Tardos, J., Cadena, C., Lopez, D., and Neira, J. (2010). Robust place recognition with stereo cameras. In *International Conference on Intelligent Robots and Systems (IROS), IEEE, Pages 5182-5189*.

Ricardo, C. and Pellegrino, S. (2010). An experimental evaluation of algorithms for aerial image matching. In *IWSSIP 17th International Conference on Systems, Signals and Image Processing*.

Siegwart, R., Sabatta, D., and Scaramuzza, D. (2010). Improved appearance-based matching in similar and dynamic environments using a vocabulary tree. In *IEEE International Conference on Robotics and Automation(ICRA), Pages 1008-1013*.

Tuytelaars, T., Bay, H., and Gool, L. (2008). Surf speeded up robust features. In *Computer Vision and Image Understanding (CVIU), Pages 110-346*.