Speeded-Up Bag-of-Words Algorithm for Robot Localisation through Scene Recognition

Tom Botterill[1,2], Steven Mills[2], Richard Green[1]

[1]Department of Computer Science, University of Canterbury, Christchurch, New Zealand.
[2]Geospatial Research Centre (NZ) Ltd., Private Bag 4800, Christchurch, New Zealand.
Email: tom.botterill @ grcnz.com

Abstract

This paper describes a new scalable scheme for the real-time detection of identical scenes for mobile robot localisation, allowing fast retraining to learn new environments. It uses the Image Bag-of-Words algorithm, where images are described by a set of local feature descriptors mapped to a discrete set of 'image words'. This scheme uses descriptors consisting of a combination of a descriptor of shape (SURF) and a hue histogram, and this combination is shown to perform better than either descriptor alone. K-medoids clustering is shown to be suitable for quantising these composite descriptors (or any arbitrary descriptor) into visual words.

The scheme can identify in real-time (0.036 seconds per query) multiple images of the same object from a standard dataset of 10200 images, showing robustness to differences in perspective and changes in the scene, and can detect loops in a video stream from a mobile robot.

Keywords: Bag-of-Words, Visual Navigation, Scene Recognition, Image Search

1 Introduction

The problem of localisation and navigation for an autonomous mobile robot in an unknown environment is widely studied, as a solution would enable many useful applications [1]. Methods for building internal maps and positioning robots from them are collectively known as Simultaneous Localisation and Mapping (SLAM). Computer vision, either alone or integrated with other sensors (INS, laser range-finders, odometry) is often used to infer motion and hence position. A key part of most SLAM implementations is loop closure [2]: this involves detecting when the robot is in a previously visited location by identifying landmarks that have been mapped previously. Errors that have accumulated in internal maps can then be corrected.

Originally loop closure detection relied on the accuracy of the current position estimate, however recently techniques from object recognition have been used to recognise when the robot is in a previously visited location without needing an estimate of the current position. This is useful as we can re-localise even when the robot is completely lost– robots navigating using vision alone can easily become disorientated due to mismatched image features, featureless environments, or due to motion blur. However, if a loop is closed incorrectly, the internal map can be corrupted.

The original Bag-of-Words algorithm is a popular algorithm for document classification, where documents are described (and categorised) by the set of words that they contain, and their frequencies. The Image Bag-of-Words algorithm was developed from this for object recognition: images are described by the set of features they contain. Recently it has been used for the recognition of scenes: Sivic et al. [3] use it to segment movies into different locations, and Níster and Stewénius [4] describe a fast and accurate implementation allowing real-time searching of image databases. These implementations both require a slow training procedure before recognition can begin. Angeli's [5] implementation allows new locations to be learned as they are encountered, but real-time (1Hz) performance is limited to about 2000 images. This paper describes our improved Image Bag-of-Words implementation which can learn to recognise scenes reliably within seconds of them first being encountered, and can search all previous scenes (ten thousand or more) in real-time (0.036 seconds per query), identifying those that appear the same.

To test and refine our method, we use Níster and Stewénius's [4] dataset, which consists of four images of each of 2550 objects (Figure 1). A system that can reliably identify the four images of the same object given one of the images should be

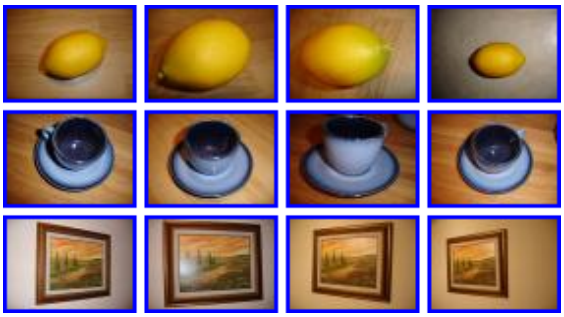

Figure 1: A sample of pictures from the test dataset

able to identify scenes that have been seen before. Scores given in this paper are the average number of correct matches out of the top four returned.

1.1 Applications

Potential localisation applications are not limited to SLAM: they include location detection for handheld or wearable computers, which may associate locations with information from previous visits, or detect scenes where Augmented Reality graphics may be inserted. Alternatively it could be used for localisation using a camera phone, with the current cell limiting the search space.

2 Approach and Relation to Existing Implementations

The Image Bag-of-Words algorithm allows us to measure the similarity of two images rapidly. In this section we describe how different parts of it work and the implementation choices we have made to enable its use for fast robot localisation.

2.1 Image Bag-of-Words Algorithm

Given a set of descriptors representing some images, choose a subset of N of them (a 'dictionary'). All descriptors can now be mapped to the closest word in this dictionary, enabling an image to be described by the N-dimensional vector of word-frequencies. Images are compared by comparing these vectors. The total frequency of a particular word is used as a measure of its distinctiveness: a rare word occurring in two images strongly suggests the images contain the same object.

Some SLAM implementation [2, 6, 7] detect loops by brute force comparisons of descriptors, but this is slow despite the small number of images involved (typically a few hundred), and further limiting of the image search space is required for real-time performance. A Bag-of-Words approach can be made hugely faster than a brute-force approach because distances between descriptors are only ever compared when we look up a word in the dictionary. Sivic et al. [3] show that brute force comparisons work no better than Bag-of-Words, as Bag-of-Words takes into account the distinctiveness of features.

2.1.1 Classifiers

A wide range of classifiers for comparing these vectors have been used, but many (such as Support Vector Machines [8] and Latent Dirichlet Analysis [9]) are aimed at categorisation rather than recognition, and training a classifier for each image would be much too slow for our application. Naïve Bayes classifiers, or direct weighted vector comparison are suitable for rapid comparisons without training. We found that the direct weighted vector comparison used by Níster and Stewénius [4] easily outperforms the Bayes classifier, or unweighted vector comparison. The vector of word frequencies is weighted by the log of the words' relative frequencies (proportion of all words seen), and vectors are compared using the L_1 norm.

2.2 Features

Features are points of interest in an image: either a random scattering of points, or 'distinctive' points (blobs or corners) that are likely to be repeatable [10]. Sampling the same feature set from two images using a distinctive point detector appears to be a good strategy, however many authors [9, 11, 12] have shown that random points are as good or better for object recognition. While existing scene recognition implementations usually use a blob detector [3, 4], we have found that random points perform better, and save the computational cost of extracting corners.

2.3 Invariant Descriptors

An invariant descriptor is some data describing a patch around a feature in an image. Two patches can be compared using the 'distance' between them; patches that are close together appear similar, and hence may be the same object. Ideally this will be true regardless of the orientation, scale, position (to a few pixels), or illumination, hence these descriptors are described as invariant. It is also desirable for descriptors to be distinctive, that is features with dissimilar appearance have distant descriptors, therefore we will use descriptors of both colour and shape.

2.3.1 Appearance/Shape Descriptors

Common descriptors of local appearance are SIFT (Scale-Invariant Feature Transform) [13] and SURF (Speeded-Up Robust Features) [10]. Each encodes a blurred patch around a feature as a vector of frequency components. Image bag-of-words implementations for scene recognition generally use

Figure 2: These hue-histograms appear similar, but are far apart if compared using the Euclidean distance.

128D SIFT vectors [4, 5, 8, 9, 14], however SURF descriptors have been shown to out-perform SIFT in both speed and accuracy for image correspondence extraction [10], and for robot localisation [15]. The shorter 36D SURF descriptor performs almost as well as the standard 64D one for our application so we have chosen to use these.

2.3.2 Colour Descriptors

A popular descriptor of colour is the hue-histogram. These are used by Filliat [14] as a descriptor for robot localisation, where they are an independent aid to localisation using SIFT features. In this paper we use hue-histograms of a small patch around each feature (typically 11×11 with a Gaussian weighting function). We have found they greatly improve the recognition results over SURF alone, while being considerably faster to compute ($3 \times 10^{-5}s$ vs. $5 \times 10^{-4}s$ for SURF descriptors).

A range of metrics are used for computing distances between histograms. Rubner et al. [16] compare several methods and find the Earth Mover's distance is best; however this is complex to compute. Jeffrey divergence is found to outperform other simple metrics, with the χ^2 metric performing almost as well. Androutsos et al. [17] compared a different range of simple metrics and found the cosine distance performed best, with the L_1 and L_2 norms performing reasonably well. These papers both compare whole-image histograms however. Filliat [14] uses the χ^2 metric for comparing histograms of patches. We have found that this and the L_2 norm have very similar performance, and outperform several other simple norms (Table 1). Those that performed poorly probably did so because many of our bins are empty (as a result of using small patches), and these are designed for the comparison of strictly positive functions.

2.4 Building the Dictionary

To initialise (or re-train) a Bag-of-Words implementation a set of representative descriptors must be chosen as our 'dictionary', and all of our images must have their descriptors encoded as these 'words'. Completing this quickly is key to learning new environments as we encounter them.

Table 1: The χ^2 and L_2 metrics performs best for hue-histogram comparison. Metrics are scaled to the same range, and combined linearly with the score from comparing SURF descriptors, giving each similar importance.

Metric	Score (top 4)
Euclidean (L_2)	3.25 ± 0.01
χ^2	3.24 ± 0.04
L_1	3.15 ± 0.00
Sum of squared diff.	3.15 ± 0.06
Max (L_∞)	2.25 ± 0.02
Jeffrey's	1.92 ± 0.08
Cosine	1.87 ± 0.01
Bhattacharyya	1.51 ± 0.04

2.4.1 Hierarchical Approach to Dictionary

A simple approach for mapping n descriptors to k words requires nk comparisons. For the test data used in this paper, n may be 5 million and k one hundred thousand. If comparisons require 200 operations then this process could take around three hours. Ideally rebuilding a dictionary will take just a few seconds, as a background process as we explore new environments. Fortunately Níster and Stewénius [4] proposed a hierarchical dictionary where each descriptor needs to be compared to 100 or fewer centres to find the closest: the descriptors are first partitioned around a small number of words (e.g. 20), then each of these partitions is partitioned again into a sub-dictionary, and so-on. In this way descriptors can be mapped to one of hundreds of thousands of words at the bottom level with only a few comparisons.

2.4.2 K-Means Clustering

Intuitively, clustering the data and using cluster centres for our dictionary appears to be a good dictionary-building strategy. Many previous attempts [4, 8, 9] at exact image recognition using a Bag-of-Words approach have used k-means clustering (Lloyd's algorithm) for this purpose. However, it is not necessarily easy to achieve substantially better results than a random selection of centres (as found by Nowak et al. [11]).

Another problem with k-means is that it treats all descriptors as a vector space with Euclidian distances (an optimal k-means clustering minimises the sum-of-squared differences (SSD) between vectors and their closest centre), even though this metric may be inappropriate for some descriptors, e.g. histograms (Figure 2). Lloyd's algorithm involves taking averages of sets of descriptors. The meaning (or appearance) of a feature partway between two hue histograms, with shape partway between the shapes of two features, is unclear.

Jurie and Triggs [12] proposed a fixed-radius clus-

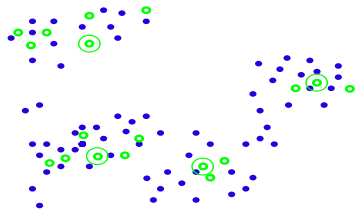

Figure 3: When a large number of points are being clustered into a small number of clusters, clustering a random subset is likely to give a good overall result.

tering algorithm which out-performs k-means for object recognition. Centres of regions of high density are repeatedly chosen from unassigned descriptors. This would be inappropriate for our top-down approach to clustering (as it involves finding dense regions rather than a good partitioning), however it would be interesting to use this for clustering at lower levels.

2.4.3 K-Medoids

K-medoids clustering allows us to cluster any set of descriptors that form a metric space. A set of representative descriptors is chosen that (usually) minimises the sum of the distances (using our metric) between descriptors and their nearest cluster centre. This allows us to cluster a set of composite descriptors with a metric consisting of any combination of metrics on the composite descriptors. Kaufman and Rousseeuw [18] describe a brute-force k-medoids algorithm (Partitioning About Medoids, or PAM) to iteratively improve on a set of centres. Each iteration has complexity $O(kn^3)$, where n is the number of descriptors and k is the number of cluster centres, and even when limiting the number of iterations (rather than stopping only when no simple refinement improves the clustering) this method becomes infeasibly slow for more than around 1000 descriptors.

2.4.4 Clustering Subsets

Even with a hierarchical dictionary we still have to cluster the entire set of descriptors at the top level. This is potentially very slow when millions of descriptors are involved. However because of the hierarchical approach used we never need to cluster into very many clusters. The CLARA (CLustering LARge Applications) algorithm, described by Kaufman and Rousseeuw [18], is ideal for this application. A random subset of a few hundred descriptors is selected and centres are chosen using the k-means or k-medoids algorithm (Figure 3). The average distance from all descriptors to the nearest centre is computed; this is what we are attempting to minimise. Further random subsets are selected and clustered, with each containing the set of centres from the best clustering found so far.

2.5 Clustering Results

Table 2 shows analysis of the clustering of 100 000 64D SURF descriptors into 21 clusters (the top-level clustering step of three with 10 000 words at the bottom level, indexing 250 images). Each test is run five times and the mean score and sample standard deviation are given. The same test was run three times for each of two other subsets of the test images and have obtained very similar relative performance for the different methods (although uniformly poorer recognition rates for some less distinctive images, such as flowers). All words and all descriptors are used in comparisons.

Performance is compared to a choice of random descriptors as centres. It is possible that all clustering is doing is finding a set of descriptors distributed throughout the space, so we also test a set of random descriptors chosen with a minimum separation constraint. However this does not make any significant difference.

We have found that all clustering methods give significantly better recognition performance than random centres (at the 1% significance level using Welch's t-test; although this assumes the results are approximately Normal, which is not necessarily valid for heuristic algorithms–we don't know how often we come across particularly bad clusterings). K-means gave the best performance; this is significantly better than any other method (5% significance level). CLARA k-means and CLARA k-medoids give clusterings of similar quality.

When it is important that clustering is completed within seconds so that a robot can learn to recognise an unknown environment, CLARA k-means or CLARA k-medoids should be used. Few iterations are necessary. For stand-alone image search applications where most or all images are available in advance, it is worthwhile to apply k-means clustering to the entire training set. This takes just over 30 minutes for the 5 million descriptors extracted from the standard dataset.

The dispersion of a clustering is the mean distance from a descriptor to its nearest centre. It is commonly used as a measure of clustering quality, however it appears to have little effect, as random clusterings have similar dispersion to k-medoids.

Processing requirements limit the PAM algorithm to around 1000 descriptors when used alone, however for CLARA k-medoids (or CLARA k-means) computation times are linear in the number of descriptors, and memory requirements are constant in the number of descriptors. For CLARA k-means we use subsets of size 1000; for k-medoids we use subsets of 120. These are chosen so that clustering takes around 0.5 seconds in each case. Doubling the sizes appears to give no significant bene-

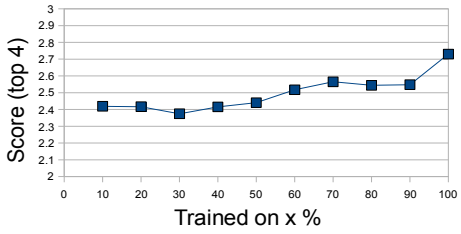

Figure 4: Benefits of re-training when new environments are encountered.

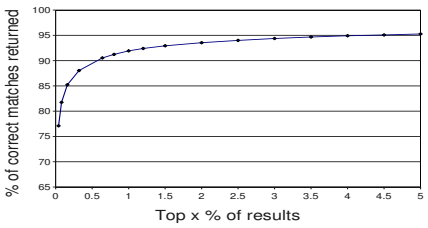

Figure 5: 95% of correct matches are returned within the top 5% of results.

fit. It is unclear whether a significant improvement is gained by increasing the number of CLARA iterations. For sets of over 250 000 descriptors most time is spent assigning all of the descriptors to centres, not for clustering subsets.

Note that this test is for one experimental setup, and there are many model parameters that may affect these figures (such as the decision to use 64D SURF descriptors). It is worthwhile trying different clustering methods to find the best for a particular application.

2.6 Retraining

To test if re-building the dictionary is worthwhile (rather than clustering once and re-using words) we have compared recognition performance on the remaining images when training is done on only the bottom few percent of the images. Figure 4 shows that training on the entire set gives a clear performance improvement, hence re-training when new environments are encountered is worthwhile.

2.7 Implementation Details

36D SURF descriptors are stored internally as 36 one byte signed integers. Histograms have 16 hue bins and 12 saturation bins, stored as 28 one-byte unsigned integers. This gives a compact representation allowing us to store 35 million descriptors in 2GB of memory, however there is a small performance hit compared with using 4 byte integers (due to byte-alignment), and rounding errors mean we cannot use a fast dot-product to compute Euclidean distances between unit vectors.

Clustering done in-place with one-byte integers is faster and does not perform significantly worse than clustering using floating point. Almost all arithmetic uses integers, so is suitable for embedded processors in handheld devices.

2.8 Parametrisation

We tuned 14 model parameters (classifier, relative significance of shape vs. colour, histogram thresholds, number of words, minimum and maximum numbers of descriptors allowed per word) with a simple genetic algorithm. Our best parametrisation rejected words with fewer than 4 or more than 20 descriptors, leaving about 100 000 words, and gave histograms about twice the importance of SURF descriptors.

3 Results and Conclusions

A system capable of accurately detecting images of the same object has been developed. Tests on a video stream from a mobile robot show we can retrain as new environments are encountered without affecting real-time performance, and this is shown to be feasible and worthwhile. Tests on the standard dataset show it to have comparable speed and accuracy with other scene recognition schemes.

3.1 Exact Object Recognition

Our implementation retrieves 3.00 correct matches on average, in a time of 0.036s. Training using CLARA k-medoids takes 200s and gives considerably better performance than treating histograms as vectors and applying k-means clustering (taking 1800s). Our memory requirements for descriptors are much less than other implementations (64 vs. 128 bytes per descriptor), and we search the entire database on each query, avoiding the need for the large inverse file needed by other implementations. On average 95% of the four correct results are returned in the top 5% of query results (Figure 5). Níster and Stewénius [4] retrieve 3.29 correct matches on average in a time of about 0.0027s per query, however this requires long 128D SIFT descriptors, k-means on all descriptors, and a large inverse file.

Using histograms alone is surprisingly effective: we can identify 2.91 images on average. Without using histograms our method performs relatively poorly (1.62/4 on average). It is unclear why this is; the main difference in our method is the use of SURF rather than SIFT. However these composite descriptors are still faster to compute than SIFT descriptors.

Table 2: Results from clustering 100k 64D SURF descriptors into 21 clusters:

Clustering method	CLARA Iterations	Dispersion	Time (s)	Score (top 4)
K-means, 20 iterations	-	39.5 ± 0.0	3.6	2.00 ± 0.05
CLARA K-medoids	6	44.2 ± 0.5	1.8	1.93 ± 0.02
CLARA K-means	1	40.2 ± 0.0	0.4	1.92 ± 0.02
CLARA K-means	6	39.9 ± 0.1	3.7	1.90 ± 0.02
CLARA K-medoids	1	44.8 ± 0.1	0.4	1.84 ± 0.07
Random set of centres	-	44.0 ± 3.1	0.3	1.71 ± 0.08
Random seperated centres	-	46.0 ± 4.1	0.9	1.71 ± 0.09

3.2 Mobile Robot Localisation

We have a 400 frame video showing three laps around and through a building. Correct scene matches are detected regularly once a lap had been completed, and many matches with the last few frames are detected, but matches are also detected before the robot had returned to a previous location. Real-time performance is maintained even when retraining in the background, and this takes only a few seconds with this many images. Our work will now focus on using our method for robot localisation and on measuring the reliability of matches.

References

[1] G. Dissanayake, H. Durrant-Whyte, and T. Bailey, "A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem," *Int. Conf. Robotics and Automation*, pp. 1009–1014 vol.2, 2000.

[2] V. Ila, J. Andrade, R. Valencia, and A. Sanfeliu, "Vision-based loop closing for delayed state robot mapping," in *Int. Conf. Intelligent Robots and Systems*, San Diego, 2007.

[3] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *ICCV*, Oct. 2003, pp. 1470–1477.

[4] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *CVPR*, Jun. 2006, pp. 2161–2168.

[5] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "Real-time visual loop-closure detection," in *ICRA*, 2008.

[6] P. E. Rybski, S. Roumeliotis, M. Gini, and N. Papanikopoulos, "Appearance-based mapping using minimalistic sensor models," *Auton. Robots*, vol. 24, no. 3, pp. 229–246, 2008.

[7] J. Kosecka, F. Li, and X. Yang, "Global localization and relative positioning based on scale-invariant keypoints," *Robotics and Autonomous Systems*, vol. 52, pp. 27–38, 2005.

[8] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[9] L. Fei-Fei and P. Pietro, "A bayesian hierarchical model for learning natural scene categories," in *CVPR*, 2005.

[10] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *ECCV*, Graz, Austria, May 2006.

[11] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *ECCV*, 2006, pp. 490–503.

[12] F. Jurie and B. Triggs, "Creating efficient codebooks for visual recognition," in *ICCV*, Washington, DC, USA, 2005, pp. 604–610.

[13] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, 1999, pp. 1150–1157.

[14] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," in *International Conference on Robotics and Automation (ICRA)*, 2007.

[15] A. Murillo, J. Guerrero, and C. Sagues, "Surf features for efficient robot localization with omnidirectional images," *Int. Conf. on Robotics and Automation*, pp. 3901–3907, 10-14 April 2007.

[16] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *Int. J. Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[17] D. Androutsos, K. Plataniotiss, and A. Venetsanopoulos, "Distance measures for color image retrieval," in *Int. Conf. Image Processing*, 1998, pp. 770–774.

[18] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, March 1990.