Visual Object Recognition using Template Matching

Luke Cole[1,2] luke.cole@anu.edu.au
David Austin[1,2] d.austin@computer.org
Lance Cole[2] lance.cole@nicta.com.au

[1]Robotic Systems Lab, RSISE
Australian National University,
ACT 0200, Australia

[2]National ICT Australia,
Locked Bag 8001,
Canberra, ACT 2601

Abstract

Template matching is well known to be an expensive operation when classifying against large sets of images as the execution time is proportional to the size of the set. This paper addresses the scaling problem by reduction of the size of the set against which new images are compared. A two-tier hierarchical representation of images for each object is created, reducing the set of images to match against. The reduced set is generated via a pre-processing task of extracting areas of interest (blobs) and then reducing the number of blobs found by clustering. The results are based on a raw database of 90 classes and a total of 140,000 images each of size 680x480. From the 90 classes, over 100,000 blobs were extracted as the training set which then was reduced by over 90%. For a recognition rate of 86%, only 0.59% of this training set was examined, giving us an execution time to identify a new image in 11 seconds.

1 Introduction

The amount of visual information present in the real world is clearly immense. However, recognising objects from visual information is challenging. The world is also constantly changing, causing most objects to be at different state each time they are encountered. This may be due to change in the viewpoint of the observer or simply variations in lighting conditions. Because of this recognising an object (in a new state) from previous studies of that object (in other states) becomes difficult. Furthermore, if we have studied a number of different objects, recognising a new object becomes a matter of comparing with each of the previous studied objects. So the time required to search our studied objects may grow proportionally with the number of objects studied. Thus, the recognition problem is how to efficiently search our studied objects ignoring "noise" (e.g. due to change in viewpoint, lighting variations etc.). Consider an example task where you wish to grab a beer from the fridge. You first need to get up, find the fridge and then after opening the fridge door you are presented with a range of groceries in the fridge. You must then search for the beer. Because you like your beer you remember what it looks like, so you try to find the beer based on what you remember. However the beer is surrounded by other goods and the fridge light is not working[1]. It is clear that you will have some trouble locating the beer. Even from this simple example, the challenges of visual object recognition are evident.

For visual object recognition, a large number of views of each object are required due to viewpoint changes and it is necessary to recognise a large number of objects, even for relatively simple tasks. Clearly, a large number of images must be gathered in order to correctly classify new objects. However, if we consider a large number of images each possibly containing a object of interest, it is impractical to search through all the images to identify a new object. The computational cost would grow with the number of images and therefore would not be even close to real time. Our training is based on two-dimensional images taken with a digital camera. However, the world is three-dimensional. All of the images are projections of the three-dimensional object and do not contain information such as depth and geometry. Therefore, it is always possible to construct cases where a (two-dimensional) visual object recognition system will fail to correctly classify three-dimensional objects. However, we hope that such cases are unlikely to occur in practice and note that the problem is still worth studying as a first method for identifying objects.

[Cyr and Kimia, 2004] approached the problem of object recognition for three-dimensional objects from two-dimensional images using an aspect-graph structure formed from the similarity between views. Their results show a recognition rate of up to 100% against a database of 64 objects. The recognition rate is rather impressive,

[1]Assume that there is some other light source!

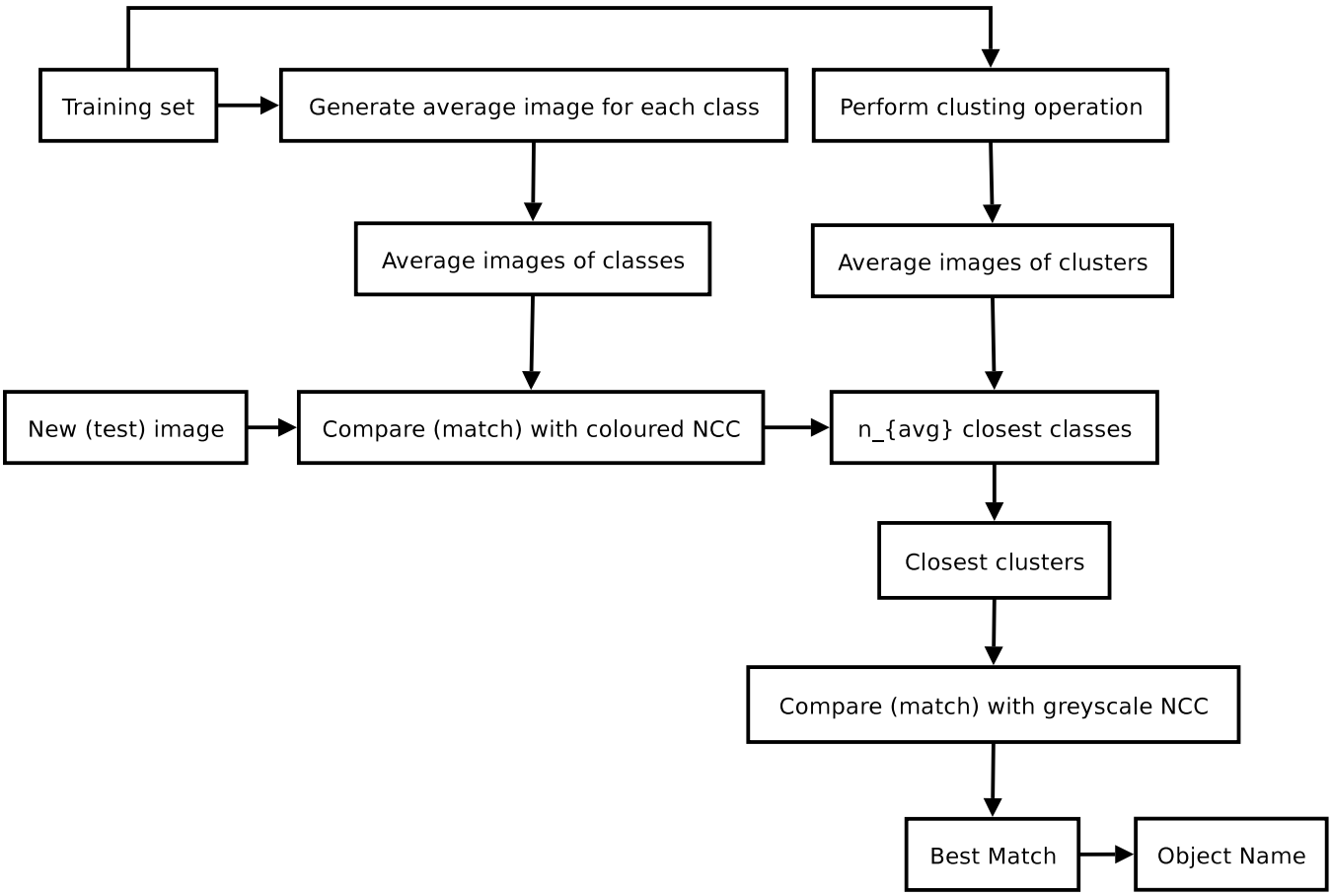


Figure 1: Simple flow chart of approach. A two-tier hierarchical approach is used: first the new image is compared with average images for each class and, second, it is compared with clustered images for some of the classes.

however the object database is rather small so we cannot be sure how the results scale to a large set of images. Other approaches for visual object recognition include template matching. [Gavrilla and Davis, 1994] proposed a modified cross correlation to allow a number of comparisons to run in parallel. Approximately 20 images could be successfully compared without loss of accuracy. Obviously, this improves the execution speed for comparing images however, the computation time still scales linearly with the size of the set. Recently, [Bala and Cetin, 2004] used decimated wavelet transforms to develop an affine invariant function for modelling of outlines of objects (model plane images). The work is an extension from previous work [Guggenheimer, 1963], [L Arbter. W. E. Synder and Hirzinger, 1990], [Reiss, 1991], [Khalil and Bayoumi, 2001], [Tieng and Boles, 1997], using affine invariant functions to recognise objects. Their results show an improvement by 47% for computational complexity without decreasing recognition performance or accuracy. However, these experiments are on a very limited test set of 20 images and only make use of the silhouette of the object. Another recent development has been the Scale Invariant Feature Transform (SIFT) [Lowe, 1999] which has also been applied to recognition of a set of ten different household objects by Ke and Sukthankar [Ke and Sukthankar, 2004]. The method uses principal components analysis (PCA) to reduce the SIFT descriptors and make them more distinctive. They report a recognition rate of 68% if the top two matches are used. Recognition takes about 5 seconds. Again the test set is very limited, though large variations of viewpoint are present.

Template matching is a simile task of performing a normalised cross-correlation between a template image (object in training set) and a new image to classify. This paper proposes a new method of using template matching across a large set. Section 2 discusses the approach to object recognition, while Section 3 explains how to reduce the large set in order to perform efficient object recognition. The results for object recognition using template matching against our reduced set are presented in Section 6.

```
function ReduceClasses(classes, M) {

  foreach (class in classes) {

    blobs = GetBlobsFor(class);
    clusters = array.empty();

    foreach (blob in blobs) {

      [blob_found, score, score_index] = NCCBlobsForBestMatch(blob, clusters);

      if (score < M)
        clusters.append(blob_found);
      else
        AddImageToAvgImage(blob_found, score_index, clusters);
    }
    reduced_classes.append(clusters);
  }

  return reduced_classes;
}
```

Figure 2: The algorithm for clustering the set of training images.

2 Approach

The motivation of this work is to visually recognise objects in real time for a large number of objects. Most existing object recognition systems suffer from poor scaling with the number of recognisable objects. Either the computational cost rises dramatically or the recognition rate drops off as the number of objects increases. For this reason, most existing approaches have only been demonstrated on small (≈ 10) data sets.

Rather than developing a more complex method for solving the problem, we wish to study a simple technique: template matching. Even limited success would mean that we have a useful technique that can be combined with other methods for higher performance. Furthermore, normalised cross correlation uses all of the information captured for an object (which is not the case for many "higher-order" recognition techniques such as [Cyr and Kimia, 2004]).

As mentioned above, we can only test the incoming image against a small subset of the training data. Figure 1 shows a graphical representation of the approach. We reduce the number of required comparisons in two ways: compare first against the average image to reduce the number of possible classes and clustering to reduce the number of comparisons within a class.

3 Training Database Reduction

Suppose we have a large set of labelled training images. Clearly, we can't classify each new test image by comparing with the whole training set. Such a brute-force approach is computationally infeasible. Hence, we wish to find a reduced set of training images for comparison. For a large training set, we can expect that a significant number of the images will be similar, due to symmetry of the object and similarity of appearance for small changes in orientation. Also, due to the nature of large training sets, we can expect a number of noisy or incorrect images, which should be treated as outliers. Furthermore, after preprocessing steps (e.g. rotation to align major axes), the training database can be expected to contain greater redundancy.

One goal here is to develop a somewhat hierarchical process, by which the required number of comparisons can be significantly reduced. So the first step that we take is to compute an average image avg_c for each of the different classes within the training set. Then when we are classifying a new image, we first compare with the set of average images and determine the n_{avg} closest matches. The closest matches are then further investigated to find the best match and hence to classify the new image.

To further reduce the number of images required for classification, we cluster images together. Clearly, finding the optimal solution to the problem of clustering a large set of images is infeasible. Hence, we adopt an *ad hoc* approach to clustering. Initially, we start with an empty set for the reduced set. Then for each image in the training set, we find the closest image in the reduced set (ranked by NCC score). We check to see if the closest NCC score is greater than a threshold M. If it is, we decide that the two images belong in the same cluster. During the process (and at the end) each cluster is rep-

```
function RecogniseObject(classify_blob, blobs) {

  best_score = -1.0;
  best_blob = NULL;

  similar_classs = ColourNCCForListOfSimilarBlobs(classify_blob, blobs);

  foreach (similar_class in similar_classes) {

    blobs = GetBlobsFor(similar_class);

    [score, score_index] = NCC(classify_blob, blobs);

    if (score > best_score) {
      best_score = score;
      best_blob = blob;
    }
  }

  return GetBlobName(best_blob);
}
```

Figure 3: The two-tier hierarchical recognition algorithm.

resented by the average of the images in the cluster. If the best NCC score is less than the threshold, then we decide that the image belongs to a separate, new cluster and initialise a new cluster in the reduced set. The algorithm is presented in Figure 2.

4 Recognition Procedure

For robotics applications, it is not acceptable to pause for long periods trying to recognise an object. Hence the focus of this work is on real-time methods and scaling to large sets of objects, more than accuracy. The recognition procedure is closely tied to the approach used above to reduce the number of images that need to be compared for recognition.

A two-tier hierarchical classification approach is used. The first step is to make an approximate classification in the form of recording possible matching classes. Given a new image we match with each of the class average images (avg_c). We then form a list of the best n_{avg} matching classes ranked by NCC score. Here we use colour information in the matching.

From the n_{avg} possible classes we investigate further to get a more accurate classification. The new image is matched against all of the cluster averages. The best match is returned as the classification for the image. Here we have used a simple greyscale NCC. We have encountered problems performing a proper colour-based comparison. Representation and working with colour information is notoriously difficult [ITB CompuPhase, 2004], [Austin and Barnes, 2003]. For the Lego blocks, using the colour information is obviously quite trivial, but we have sought to develop a more general, rigorous approach. Unfortunately, this has failed, so we have had to resort to a more Lego-specific approach: greyscale NCC to identify the shape and average colour comparison to determine the block colour (we call this the colour hack approach). This is a weakness of our test data set: the colours are particularly distinct.

5 Object Database

The data set chosen to train and classify from, was images of Lego bricks. The database is large both in number of classes (89), and number of examples per class (at least 1000), for a current total of 100,000 images. Figure 4 shows an example image of each Lego brick tested. The scale of the data set allows for thorough testing of performance (particularly scalability) both in terms of preprocessing/learning, and recognition. Also, the difficulty of recognition increases with database size.

To give the database the desired features, the method for acquiring images was somewhat unconventional. The camera was in a fixed position, and each Lego brick was repeatedly dropped from a height of about 15cm to fall on a platform in a random pose. Obviously, the machine had to be made out of Lego itself.

For each raw image of a Lego brick the blob has be cropped out of the image and masked. The cropped blob is then rotated to its main axis then the object's centre of mass is translated to the middle of the image. Figure 4 shows an example cropped raw image of each Lego brick class tested.

Figure 4: Example images of each of the Lego bricks in the object database.

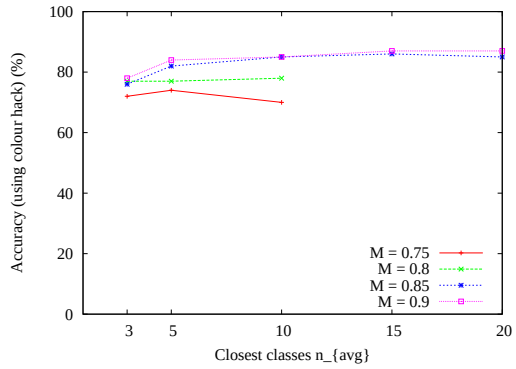


Figure 5: Recognition accuracy vs number of similar views (n_{avg}) for recognising 1000 new images against various reduced sets (M). (Using the colour hack.)

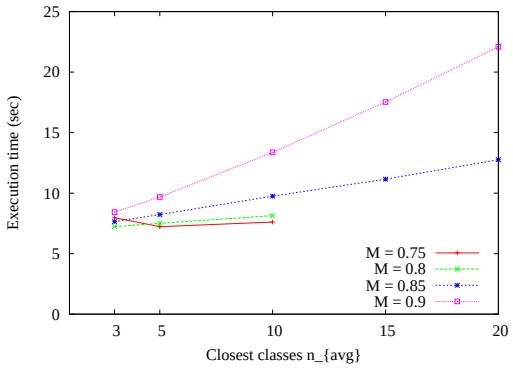


Figure 6: Execution time per recognition vs number of similar views (n_{avg}) for recognising 1000 new images against various reduced sets (different values of M).

6 Results

The success of the algorithm was tested via comparing the trade-off of different sized reduced sets and the number of similar classes (n_{avg}) for better accuracy and execution time. As expected the greater the number of images examined by the algorithm (due to different values of M and n_{avg}), the better the results.

Figure 13 presents the average images for each class. The blurring of the images is caused by changes in appearance from different viewpoints. From the lack of blurring for some classes, it can be seen that they have few (or even only one) appearance. For example, the `holeconnector-grey-5` (sixth column, third last row) has only one characteristic view.

The recognition rates as parameters are varied are shown in Figure 5. The best recognition rate is 86% for $n_{avg} = 15$ and $M = 0.85$ (we discount the marginally better performance for $M = 0.9$ because of the significantly higher computation time). While this recognition rate is not fantastic, it is not too bad, given the simplicity of the approach and compares well with the state-of-the-art approaches discussed in the introduction.

Figure 6 shows the variation in recognition time for recognition for a range of different values of M (and, hence, different size of clustered sets). The results are obtained on a AMD Athlon(tm) XP 2700+ processor, with 1GB of memory, running Linux. The algorithm was implemented in C, without much optimisation. The execution time is a little longer than we would like, however this method still compares favourably with more complex methods that take many minutes for recognition [Cyr and Kimia, 2004]. The times ($10 - 12$ seconds) are not quite suitable for real time. However, applying techniques such as [Gavrilla and Davis, 1994] could permit a speed up of 10 to 20 times. Furthermore, considerable optimisation of the NCC implementation is possible.

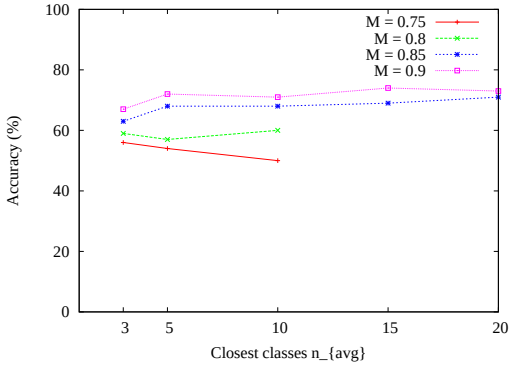


Figure 7: Average accuracy vs number of similar views (n_{avg}) for recognising 1000 new images against various reduced sets.

Figure 7 shows the recognition rates as parameters are varied in the case that we use a simple greyscale NCC for the second phase comparison (i.e. not using the colour hack to find the closest colour). The recognition rates are much lower because it often confuses bricks of different colours. This is not surprising since the greyscale NCC will neglect intensity shifts caused by the different colours. Thus noise will perturb the matching process.

The number of clusters for each class is shown in Figure 8. It can be seen that the clustering process has dramatically reduced the number of images that we have to store and potentially compare against. Figure 9 shows the number of images examined per match. It can be seen that the proposed method uses a very small percentage (0.59%) of the original training set.

To gain insight into the approximate and more accurate classification techniques used here see Figures 10 and 11 which show (far left) the average image of all angle views for `anglebracket-grey-2x2` and `block-black-4x2`, respectively. For our data set there

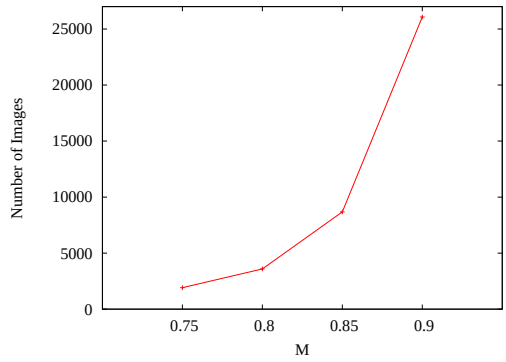


Figure 8: Number of clusters for each reduced set from training set of 101,936 images.

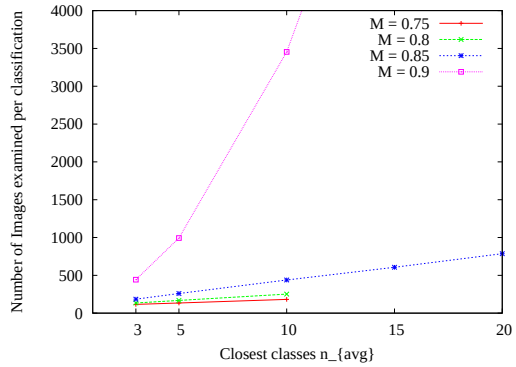


Figure 9: Average number of images examined from each reduced set per classification vs. similar views (n_{avg}) for recognising 1000 new images against various reduced sets.

are a total of 90 of these images which are used in the first phase approximate classification. The other images in Figure 10 and 11 show the same view obtained from different reduced sets. They illustrate how the algorithm sees the similarity between views and as expected the higher the threshold M the fewer views are used in creating a cluster (average) image.

7 Discussion and Future Work

This paper has presented a simple method for visual object recognition. Visual object recognition is a hard problem and cannot really be solved (from a single view there are often ambiguities). Furthermore, to process the necessary data fast enough for a real-world application is also difficult. The simple template matching algorithm presented here has achieved promising results with 86% recognition rates in times of 11 seconds. However, further work is required before it will be suitable for integration into another system.

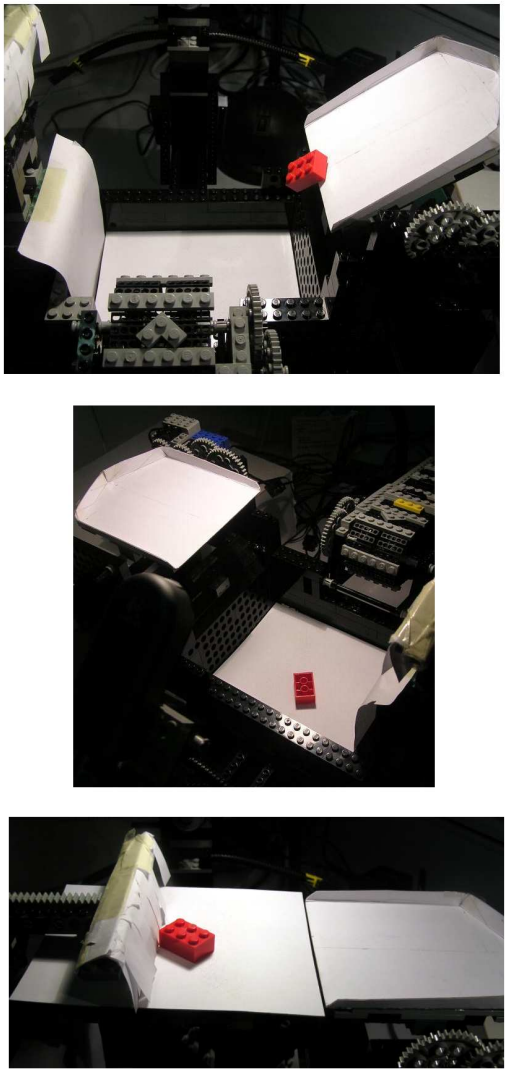
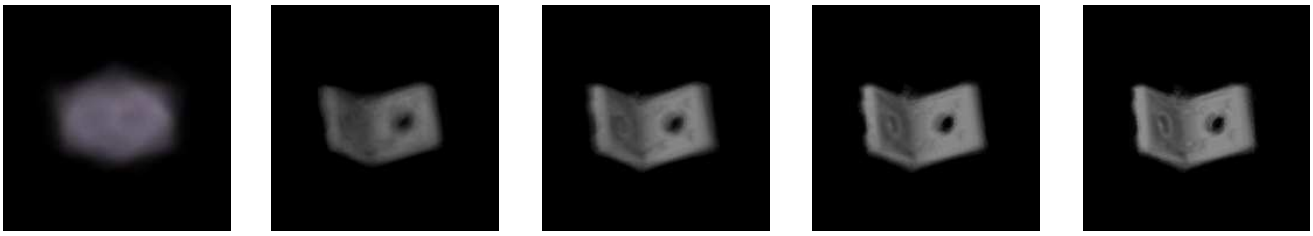


Figure 12: The data acquisition process: first dropping a piece, then taking a picture, then brick is pushed off the (raised) platform so the process can repeat.

The largest weakness of the method is the *ad hoc* approach to clustering. A significant improvement would be to replace the approximate clustering technique using a simple threshold (M) with a more rigorous method. Also, the use of colour in matching needs work.

8 Acknowledgements

This work was supported by funding from National ICT Australia. National ICT Australia is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence program.



Average image (1024 images) | Cluster for $M = 0.75$ (129 images) | Cluster for $M = 0.8$ (57 images) | Cluster for $M = 0.85$ (31 images) | Cluster for $M = 0.9$ (22 images)

Figure 10: Example class average and one cluster average for a range of different thresholds M for brick `anglebracket-grey-2x2`.

Average image (991 images) | Cluster for $M = 0.75$ (319 images) | Cluster for $M = 0.8$ (217 images) | Cluster for $M = 0.85$ (59 images) | Cluster for $M = 0.9$ (45 images)

Figure 11: Example class average and one cluster average for a range of different thresholds M for brick `block-black-4x2`.

References

[Austin and Barnes, 2003] D. Austin and N. Barnes. Red is the new black - or is it? In *Australasian Conference on Robotics and Automation*, December 2003.

[Bala and Cetin, 2004] E. Bala and A. E Cetin. Computationally efficient wavelet affine invariant functions for shape recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1095–1099, August 2004.

[Cyr and Kimia, 2004] C. M. Cyr and B. B. Kimia. A similarity-based aspect-graph approach to 3d object recognition. *International Journal of Computer Vision*, 2004.

[Gavrilla and Davis, 1994] D. M. Gavrilla and L. S. Davis. Fast correlation matching in large (edge) image databases. In *Proc. of the 23rd AIPR Workshop*, August 1994.

[Guggenheimer, 1963] H. W Guggenheimer. *Differential Geometry*. McGraw Hill, 1963.

[ITB CompuPhase, 2004] ITB CompuPhase. Colour metric, 2004. http://www.compuphase.com/cmetric.htm.

[Ke and Sukthankar, 2004] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition*, 2004.

[Khalil and Bayoumi, 2001] M. I Khalil and M. M. Bayoumi. A dyadic wavelet affine invariant function for 2d shape recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1152–1164, October 2001.

[L Arbter. W. E. Synder and Hirzinger, 1990] H. Burkhardt L Arbter. W. E. Synder and G. Hirzinger. Application of affine-invariant fourier descriptors to recognition 3-d objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):640–647, July 1990.

[Lowe, 1999] David G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision, Corfu, Greece*, pages 1150–1157, 1999.

[Reiss, 1991] T. H. Reiss. The revised fundamental theorem of moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):830–834, August 1991.

[Tieng and Boles, 1997] Q. M. Tieng and W. W. Boles. Wavelet-based affine invariant representation: A tool for recognizing planar objects in 3d space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):846–857, August 1997.

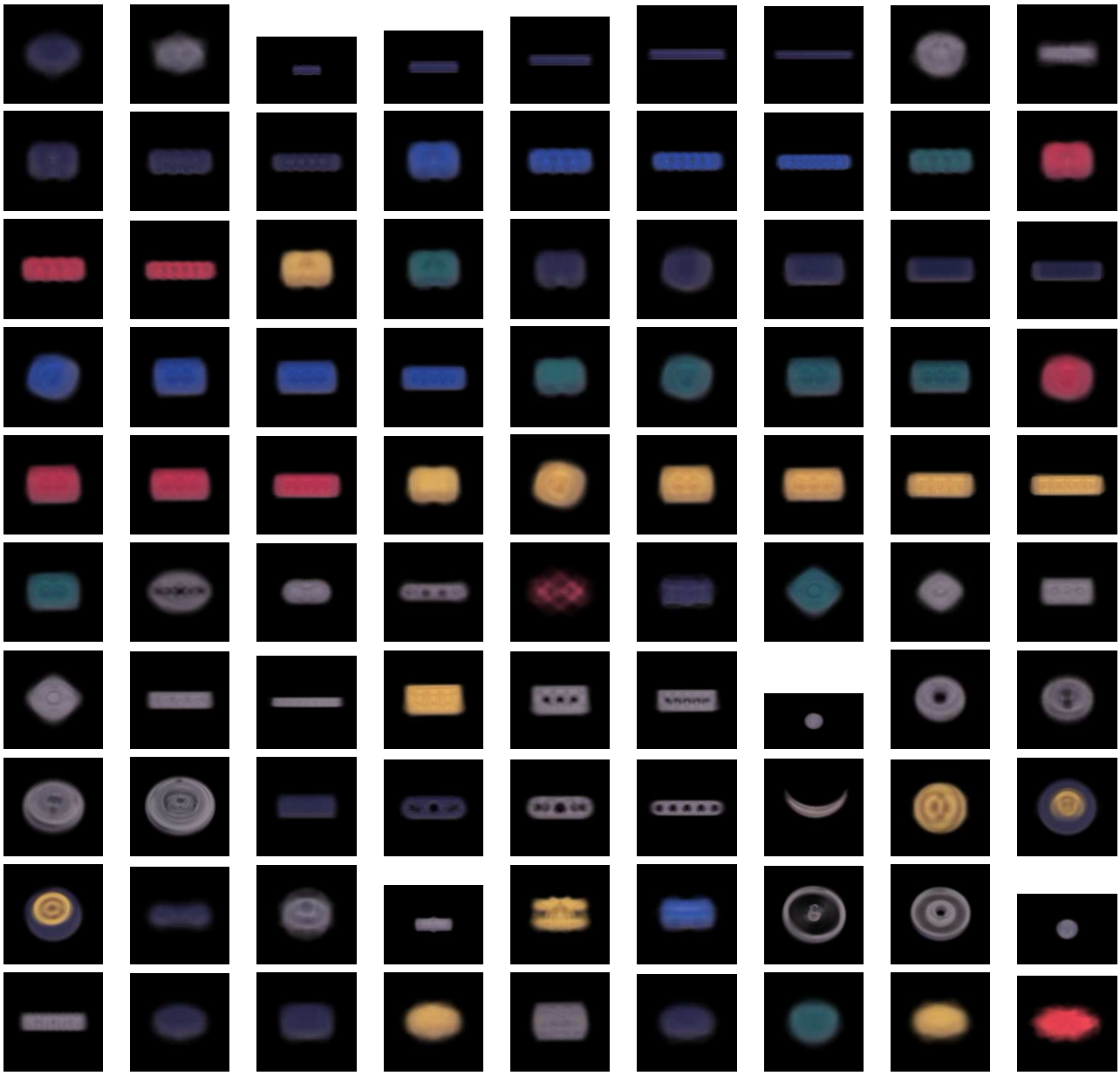


Figure 13: Average images of each of the Lego bricks in the object database.