Automatic Selection of Optimal Views in Multi-view Object Recognition

Farzin Mokhtarian and Sadegh Abbasi

Centre for Vision Speech and Signal Processing
University of Surrey, Guildford, GU2 5XH UK
[F.Mokhtarian][S.Abbasi]@surrey.ac.uk
Tel:+44-1483-876035 Fax:+44-1483-876031
http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html

Abstract

A shape-based method for multi-view 3-D object representation and recognition is introduced and explored in this paper. 3-D objects are recognised by a small number of images taken from different views. The paper addresses the issue of automatic selection of the best and the optimum number of views for each object.

The object boundary of each view is considered as a 2-D shape and is represented effectively by less than ten pairs of integer values. These values include the locations of the maxima of its Curvature Scale Space (CSS) image contours. The CSS shape descriptor is expected to be selected for MPEG-7 standardisation. An unknown object is then recognised by a single image taken from an arbitrary viewpoint.

The method has been tested on a collection of 3-D objects consisting of 15 aircrafts of different shapes. Each object has been modelled using an optimised number of silhouette contours obtained from different view points. This number varies from 5 to 25 depending on the complexity of the object and the measure of expected accuracy. A comprehensive analysis of the performance of the system has been given in this paper as the number of views varies.

Around ten silhouette contours corresponding to random views are separately used as input for each object. Results indicated that robust and efficient 3-D free-form object recognition through multi-view representation can be achieved using the CSS representation even for large database retrieval applications.

1 Introduction

There are two major methods of data acquisition in 3-D object representation. In laser-based systems, range data is produced which essentially records the distances between the camera and different points of the object. In image-based systems, a CCD camera is

BMVC 2000 doi:10.5244/C.14.28

used to produce 2-D images from the object. In both methods, the 3-D object is finally represented by a number of features extracted either from the range data or from 2-D images. While the range data provides more accurate information about the surface of the 3-D object, the laser-based systems are more expensive and the related recognition methods are more time-consuming. The computational cost of a method has recently become much more important in search and retrieval from large databases. Although hierarchical methods, and different indexing techniques have been introduced to narrow down the search space in such applications, the need for rapid matching methods to find the best matches among the remaining candidates still exists.

A large number of 3-D object representation methods have been introduced in the literature. They can be categorised based on the data acquisition techniques or the type of descriptors they extract from this data to represent the 3-D object. It is also to be mentioned that some methods impose certain restrictions on the classes of geometrical objects that can be handled.

In [11], parallel lines and ellipses are used to describe different view points of an object. A strategy is suggested to recognise an object from an unknown viewpoint. The method is based on earlier works by Brooks [6] which has been modified later on by others [9][10]. In [15], a model is established from a large number of viewpoints taken from a video sequence. The input of the system is also a video sequence of an unknown object. The system first builds a 2-D representation of the object. If the representation matches one of the objects of the database, it is modified based on new information extracted from the new sequence. Otherwise the object is recognised as a new one and its representation is stored. The number of views may be reduced with a more sophisticated preprocessing [5]. Representation with multiple views and recognition using a single view was further proposed in [7]. The number of viewpoints used to represent a complex object was down from about 2000 in [15] to $20 - 30$ as reported in [7]. Shape [16][8] and color [13] features have also been used in 3-D object representation. The problem with appearance-based methods [12][14] is that the technique is influenced by illumination conditions and the background.

While most 3-D object representations are complicated and inefficient, conventional multi-view representations are based on a large number of views and can not be used in many applications such as retrieval from large databases. Multi-view representations have not yet successfully dealt with the following issues:

- What is the optimal number of views?
- How to select the optimal views?

In this article, we propose a method for automatic selection of optimal views of an object. In order to represent an object efficiently, we eliminate similar views and select a relatively small number of views using an optimisation algorithm. This number varies from 5 to 25 depending on the complexity of the object and the measure of expected accuracy.

To identify an unknown object from a single viewpoint, its representation is matched with all images of the database and the best matches are retrieved and displayed. In our experiment with a collection of 15 toy aircrafts of different shapes, we observed that almost always, the first output of the system is the same object as the input query.

The maxima of Curvature Scale Space (CSS) image have already been used to represent shapes of boundaries in similarity retrieval applications [3]. The representation has been proved to be robust under similarity transformation which include translation,

scaling and changes in orientation. The representation is also robust under general affine transforms [1].

The following is the organisation of the remainder of this paper. In section 2, CSS image is reviewed and the CSS matching is briefly explained. Section 3 is devoted to the application of the method in multi-view object representation where the optimisation algorithm and related experiments are explained. Section 4 represents the results and concluding remarks are presented in section 5.

2 Curvature Scale Space image and CSS matching

Consider a parametric vector equation for a curve:

$$\Gamma(u) = (x(u), y(u))$$

where u is an arbitrary parameter. The formula for computing the curvature function can be expressed as:

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{(\dot{x}^2(u) + \dot{y}^2(u))^{3/2}} \quad . \tag{1}$$

Equation (1) is for continuous curves. In order to calculate the curvature of a digital curve, we use the idea of *curve evolution* which basically studies shape properties while deforming in time.

If $g(u, \sigma)$ is a 1-D Gaussian kernel of width σ, then $X(u, \sigma)$ and $Y(u, \sigma)$ represent the components of *evolved* curve Γ_σ,

$$X(u, \sigma) = x(u) * g(u, \sigma) \qquad Y(u, \sigma) = y(u) * g(u, \sigma)$$

according to the properties of convolution, the derivatives of every component can be calculated:

$$X_u(u, \sigma) = x(u) * g_u(u, \sigma) \qquad X_{uu}(u, \sigma) = x(u) * g_{uu}(u, \sigma)$$

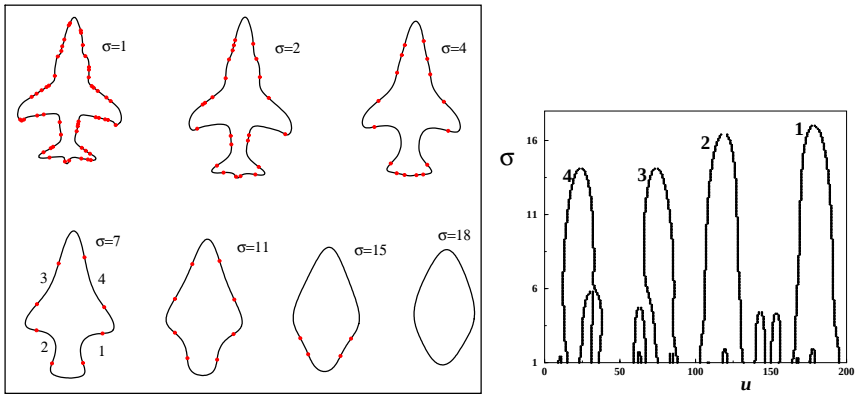
and we will have a similar formula for $Y_u(u, \sigma)$ and $Y_{uu}(u, \sigma)$. Since the exact forms of $g_u(u, \sigma)$ and $g_{uu}(u, \sigma)$ are known, the curvature of an evolved digital curve is given by:

$$\kappa(u, \sigma) = \frac{X_u(u, \sigma)Y_{uu}(u, \sigma) - X_{uu}(u, \sigma)Y_u(u, \sigma)}{(X_u(u, \sigma)^2 + Y_u(u, \sigma)^2)^{3/2}} \tag{2}$$

As σ increases, the shape of Γ_σ changes. This process of generating ordered sequences of curves is referred to as the evolution of Γ.

2.1 Curvature Scale Space image

If we calculate the curvature zero crossings of Γ_σ during evolution, we can display the resulting points in (u, σ) plane. For every σ we have a certain curve Γ_σ which in turn, has some curvature zero crossing points. As σ increases, Γ_σ becomes smoother and the number of zero crossings decreases. When σ becomes sufficiently high, Γ_σ will be a convex curve with no curvature zero crossing, and we terminate the process of evolution. The result of this process can be represented as a binary image called CSS image of the curve (see Figure 1).



(a) **(b)**

Figure 1: a)Shrinkage and smoothing of the curve and decreasing of the number of curvature zero crossings during evolution. b)The CSS image of the shape. Note that for each segment of the shape there is a contour in the CSS image.

As seen in Figure 1, the CSS image consists of several arch shapes contours, each related to a segment of the shape. For large contours, the corresponding segments are marked in this Figure. This shape is finally represented by the locations of the maxima of its CSS contours. Small contours of the CSS image are related to noise or small ripples of the curve. In order to avoid complicated and inefficient matching, small maxima are not included in the representation.

2.2 Properties of the CSS image

The CSS representation is robust with respect to scale, noise and change in orientation. A 2-D rotation of the object usually causes a circular shift on its representation which is easily determined during the matching process. Note that the effect of a change in the starting point is also the same. Noise may create some small contours on the CSS image, but the main contours and therefore the corresponding maxima remain unaffected.

Compactness is another aspect of the CSS representation. A shape is represented by about less than ten pairs of integer values which can be determined without any ambiguity. The matching algorithm which compares two sets of representations and assigns a match value as the measure of similarity between the shapes is also simple and fast.

Another property of the CSS image is that it retains the local information of the shape. Every contour of the CSS image corresponds to a concavity or a convexity of the shape. A local deformation of the shape mainly causes a change in the corresponding contour of the CSS image. Using this property, one can include more local information about the shape in the CSS image.

2.3 Curvature Scale Space Matching

The algorithm used for comparing two sets of maxima, one from the input (also called *image*) and the other from one of the models, has been described in [2]. The algorithm

Figure 2: One view of each object of the database.

first finds any possible changes in orientation which may have been occurred in one of the two shapes. A circular shift then is applied to one of the image maxima to compensate the effects of change in orientation. The summation of the Euclidean distances between the relevant pairs of maxima is then defined to be the matching value between the two CSS images.

The following is a condensed version of the algorithm which includes the basic concepts.

- Apply a circular shift to all image maxima so that the $u\_coordinates$ of the largest maxima, one set from the image and the other from the model become identical.
- Starting from the second largest maximum of the image, determine the nearest maximum of the model to each maximum of the image.
- Consider the cost of the match as the summation of the Euclidean distances between the corresponding maxima.
- If the numbers of the image and the model maxima differ, add the σ coordinates of the unmatched maxima to the matching cost.

3 Multi-view object representation

The underlying idea is to model a 3-D object using a small number of silhouette contours obtained from different viewpoints. The difficulties in implementing this idea include occlusion, transformations and segmentation.

Occlusion occurs when some parts of the objects hide behind the other parts. As a result, the 2-D boundary contours of the objects are damaged severely. The CSS representation can deal with minor occlusion as it affects the shape locally and only one or two CSS maxima may be affected. However, major occlusions must be dealt with by increasing the number of viewpoints used to represent the object. The optimum number of viewpoints depends highly on the geometry of each individual object and varies from one to another. Since each view is represented by a small size feature vector, and the matching

process is very fast, this method can be used in retrieval applications dealing with a large number of objects, each represented by a number of viewpoints.

Affine transformation is another phenomenon which occurs in multi-view object representation. It is important for the features extracted from the images to be affine invariant. It has been shown [1] that the CSS representation is robust under general affine transforms.

Segmentation of the images is necessary to extract the boundary of objects prior to CSS computation. This is not always an easy task and can not always be carried out automatically. Interactive methods [4] may be used in the case of complicated scenes. Our experiments were carried out on a collection of 15 toy aircrafts. A video sequence was prepared from each object using a 3CCD digital video camera. The color images were then grabbed from the video sequences. In order to avoid a complicated pre-processing stage and segmentation errors, the objects were put on a simple stand and illuminated by two lamps against a dark background. One view of each object of our database is represented in Fig. 2.

From the video sequence, a number of images from each object were grabbed from different views. On average, 53 images per object were grabbed at this stage. Fig. 3(a) shows all views of a particular object. In order to eliminate similar views and achieve an efficient representation, we introduced an algorithm for automatic selection of optimal views as explained in the following subsection.

3.1 Optimal view selection

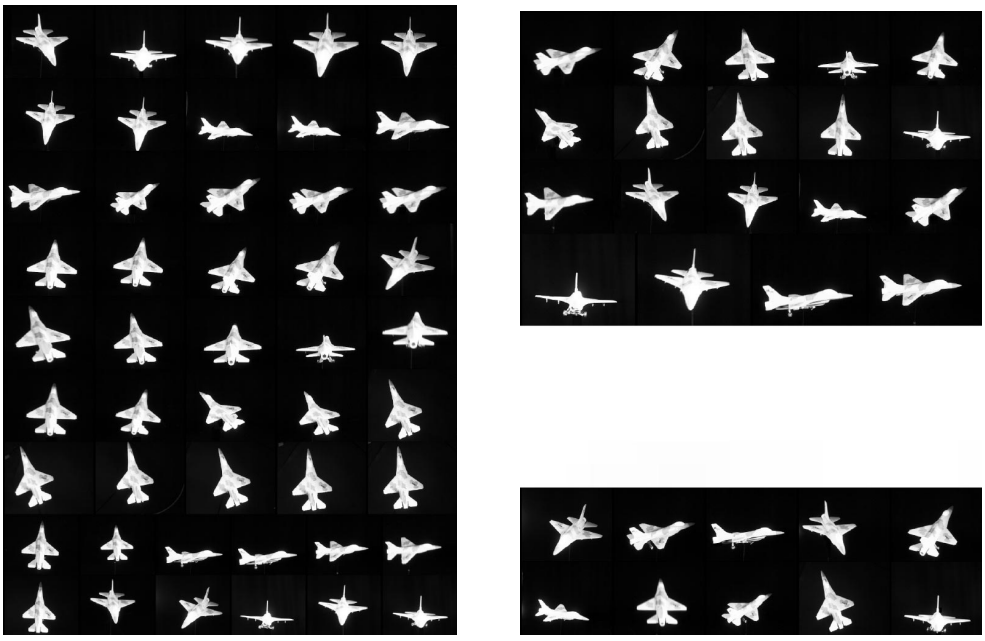
When an object is pictured from a large number of viewpoints, it is likely that some of the resulting images are similar and convey no additional information. As a result, an algorithm is required to identify the number of images needed to represent an object. We use the CSS representation and its associated matching algorithm to measure the similarity between two different images grabbed from a single object. The algorithm is as follows.

1. Obtain many views of the object from different viewpoints. The jth object will have n_j views:
$$V_1(o_j), V_2(o_j), ..., V_{n_j}(o_j)$$

2. Segment the images obtained in step 1 to recover the boundary contours. For each boundary, compute its CSS image and extract the maxima of this image to obtain:
$$CSS(V_1(o_j)), CSS(V_1(o_j)), ..., CSS(V_{n_j}(o_j))$$

3. Select a threshold value *t* which will be used to define which views are similar. If the matching cost between two views is less than *t* they are marked as similar.
$$(M\_cost(CSS(V_j), CSS(V_k)) \leq t) \Rightarrow Sim(V_j, V_k) = TRUE$$

4. Calculate the matching cost between each representation, obtained from a contour in step 1, and all other representations, obtained from other contours. If the matching cost is less than *t*, declare the two views as similar. Assign a rank *r* to each view defined as the number of views that are similar to it.
$$r(V_j) = sizeof\{V_k|\ Sim(V_j, V_k) = TRUE\}$$



(a) **(b)**

Figure 3: a)An object of our database is initially represented by about 50 views. b)Top: The reduced set of views after view selection. 19 views are selected in the case of $t = 0.3$. Bottom: Input views. 10 views are randomly selected to be used as queries.

5. Make a sorted list L of all views. Each view will have a pointer to other views similar to it.

$$L = \{V_j, V_k, ..., V_l |\ r(V_j) \geq r(v_k) \geq ... \geq r(V_l)\}$$

6. Start from the top of L and place the first view in the set C of characteristic views. Remove all views similar to the first view of L to obtain a reduced list.

7. Move down the reduced list L and repeat the procedure in 6 until the end of L is reached.

At the end of this process, the set C will contain the full set of characteristic views of the input object determined automatically. The result of this process for the object of Fig. 3(a), is presented in top of Fig. 3(b).

Note that the algorithm involves the matching of each view of an object with all other views of the same object, and it must be applied to all objects. In the case of large databases, the process may be time consuming; however, it is done off-line and has no effects on the recognition stage which compares the input query to all selected views of different objects.

t	0	0.1	0.25	0.3	0.35	0.4	0.5	0.8	1.0
n_a	53	38	23	20	17.5	15.6	13	7.7	5.7
S.R. (%)	97	97	98	98	96	94	93	74	67

Table 1: The success rate, S.R. and average number of images per object, n_a, for different values of threshold value t.

3.2 Recognition experiment

After selecting the optimal views of each object, the related representations of them, consisting of the locations of maxima of their CSS image contours were stored. A pointer to the related object was also stored with the representation.

In order to test the performance of the method, we recorded another video sequence from each object. A set of 10 test images were then randomly grabbed from each sequence. For the object of Fig. 3(a), this set is presented at the bottom of Fig. 3(b). After extracting the boundary of the objects and computing the CSS images of them, the maxima of these images were used to find similar images from the database produced by the automatic view selection algorithm.

Each test image was then used as an input query and the system was asked to retrieve all similar images based on the CSS representation. We observed that in almost all cases the recognition was possible as the first output of the system was another view of the same object. The results are presented in the following section.

4 Results

The key parameter which most affects the results is n_j, the number of images used to represent the jth object. This parameter varies from an object to another and therefore we use its average n_a as follows.

$$n_a = \frac{1}{M}\sum_{j=1}^{M} n_j$$

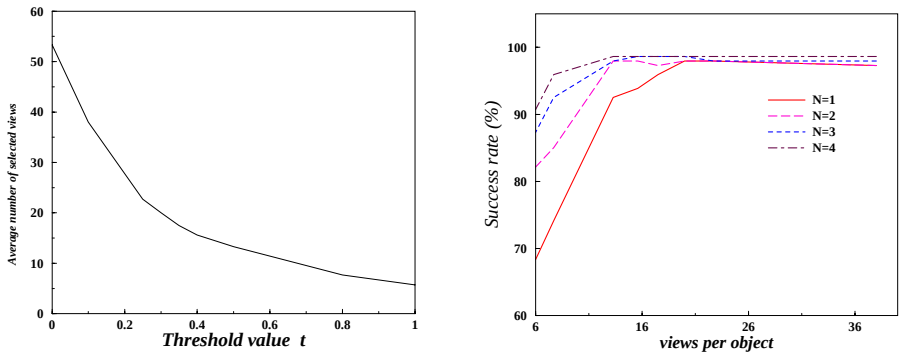
where M is the number of objects. Both n_j and n_a are controlled by the threshold value t in the view selection algorithm as seen in Fig. 4(a) and table 1.

We used different values of t and obtained different values for n_a. In general, higher n_a leads to a better results. However, it is not always the case and there is an optimum number for n_a which must experimentally be determined for each collection of objects.

In the first row of table 1, $t = 0$ means that no optimisation has been made and all initial views have been used to recognise the models.

For our collection, the best result can be achieved by considering the threshold value, t, as 0.3 and using $n_a = 20$ images per object. Under this condition, where each object is represented by almost 20 views, the recognition rate of 98% was achieved.

In Fig. 4(b) the success rate for different values of observed outputs, N, have been plotted. If at least one of the first N outputs of the system represents the same object as the input query, then the result for that query is considered as a success. For lower values of n_a, eg $n_a < 15$, if the first output of the system does not represent the correct object,



(a) **(b)**

Figure 4: a) The average number of views per object is directly related to the threshold value, t. See subsection 3.1. b) The success rate increases as the number of views per object increases. N is the number of observed outputs.

the second or third outputs are likely to do so. In the case of large databases, it is crucial for n_a to be as small as possible. The output of the system is then verified by the user who will be able to select the correct object among the first N outputs of the system.

As shown in table 1 and Fig. 4(b), even if we consider only the first output of the system, in a wide range of threshold value t, namely from 0.1 to 0.35, a very high success rate can be achieved.

Overall, the results indicate that by representing an object with several views, recognition based on the first output of the system is feasible. At the same time, in image database approach, where the user verifies the output of the system, the number of standard views can be reduced.

5 Conclusions

The Curvature Scale Space representation has already been used for applications such as shape recognition and shape similarity retrieval. It has also shown robustness under general affine transforms. In this paper, we introduced a shape-based method of multi-view 3-D object representation and recognition. Each view of the object was represented by the locations of the maxima of the CSS image of its silhouette contour. Since the optimal number of views depends on the geometry of the object, we proposed a new method for automatic selection of optimal views. We then tested the method on a collection of 15 toy aircrafts with different shapes.

Results indicated that fast and efficient recognition of 3-D objects can be achieved through CSS-based multi-view representation and recognition.

References

[1] S. Abbasi and F. Mokhtarian. Shape similarity retrieval under affine transform: Application to multi-view object representation and recognition. In *Proceedings of the 7th IEEE Interna-*

tional Conference on Computer Vision, ICCV99, pages 450–455, Kerkyra, Greece, September 1999.

[2] S. Abbasi, F. Mokhtarian, and J. Kittler. Shape similarity retrieval using a height adjusted curvature scale space image. In *Proceedings of 2nd Int. Conf. on Visual Information Systems*, pages 173–180, San Diego, CA, USA, December 1997.

[3] S. Abbasi, F. Mokhtarian, and J. Kittler. Curvature scale space in shape similarity retrieval. *Multimedia Systems*, 7(6):467–476, November 1999.

[4] A.A. Amini, T.E. Weymoth, and R.C Jain. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–866, 1990.

[5] G. Bradski and S.Grossberg. Fast-learning viewnet architectures for recognizing three-dimentional objects from multiple two-dimentional views. *Neural Networks*, 8(7/8):1053–1080, 1995.

[6] R.A. Brooks. Symbolic reasoning among 3-d models and 2-d images. *Artificial Intelligence*, 17:285–348, 1981.

[7] J. Dunker, G. Hartmann, and M. Stohr. Single view recognition and pose estimation of 3d objects using sets of prototypical views and spatially tolerant contour representations. In *Proceedings of the 13th International Conference on Pattern Recognition,*, volume 4, pages 4 – 18, August 1996.

[8] A.C. Evans, N.A. Thacker, and J.E.W. Mayhew. A practical view based 3d object recognition system. In *Proceedings of third International Conference on Artificial Networks*, pages 6–15, August 1993.

[9] C. Goad. Special purpose automatic programming for 3d model-based vision. In *Proceedings of Image Understanding Workshop*, pages 94–104, June 1983.

[10] K.S. Hong and K. Ikeuchi. Generating a strategy for configuration determination: a module of vision algorithm compiler for object localization programs. In *Reprints of the 5th International Symposium of Robotics Research*, pages 164–171, 1989.

[11] Y. Kuno, Y. Okamoto, and S. Okada. Object recognition using a feature search strategy generated from a 3-d model. In *Third IEEE International Conference on Computer Vision*, volume 5, pages 626–635, December 1990.

[12] H. Murase and S. Nayar. Three-dimensional object recognition from appearance- parametric eigenspace method. *Systems and Computers in Japan*, 26(8):45–54, July 1995.

[13] A.A.Y. Mustafa, L.G. Shapiro, and M.A Ganter. 3d object recognition from color intensity images. In *Proceedings of 13th International Conference on Pattern Recognition, ICPR'96*, volume 1, pages 627–631, August 1996.

[14] S. Nayar and H. Murase. Dimensionality of illumination in appearance matching. In *Proceedings of 13th IEEE International Conference on Robotics and Automation*, volume 2, pages 1326–1332, Minneapolis, MN, USA, April 1996.

[15] M. Siebert and A. M. Waxman. Adaptive 3-d object recognition from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:107–124, 1992.

[16] J.Y. Wang and F.S Cohen. 3d object recognition and shape estimation from image contours using B-splines, unwarping techniques and neural network. In *1991 IEEE International Joint Conference on Neural Networks*, volume 3, pages 2318–2324, November 1991.