SHAPE DETECTION, ANALYSIS AND RECOGNITION

ILYA LEVNER

August 22, 2002

ABSTRACT. This paper surveys the various techniques for shape recognition and analysis with emphasis on robustness. Specifically, a review of boundary shape analysis methods and techniques is presented followed by description and experimental results of a new technique, based on *Markov Shape Theory*, that may alleviate the problems experienced by classical methods for shape analysis and boundary extraction in the presence of various kinds of noise.

CONTENTS

1. INTRODUCTION

An important aspect of artificial perception deals shape recognition in static scenes. Having a close relationship with biological recognition paradigms, shape recognition in artificial systems has been approached in several ways. Originally proposed by [Marr76] recognition of shapes can be performed in several ways characterized by *Shape from x* research as:

- **Shape from contours**
- Shape from shading
- Shape from texture
- Shape from stereo
- Shape from fractal geometry

This paper takes a close look at a subset of "shape from contours" family of techniques, namely boundary scalar and boundary space-domain techniques. After surveying the most representative techniques within each category the paper proceeds to outline a new method for shape boundary extraction and analysis based on Markov Shape Theory. The new technique, currently dubbed Viterbi-Max-Max or VMM for short, presents a novel approach to handling shape analysis within static images containing noise, occlusions and various other perturbations presenting problems for classical methods of shape analysis. Some initial experimental results will be presented followed by a brief discussion of future work geared towards further development of Markov Shape Theory approaches as robust contour extraction and analysis tools.

1.1. Classical Shape Analysis Techniques.

In order to assert whether a shape of interest is present in an image, typically one must have a notion of a shape template to match with the shape present in an image. Over the last 3 decades various methods and techniques have been developed to enable comparison of shapes. Traditionally "shape from contours" analysis has been approached in two ways, through *shape description* techniques and *shape representation* techniques. Shape description techniques typically transform the original shape into a numeric representation of key shape characteristics, while representational techniques use non-numeric approaches to capture important shape characteristics. In addition to the aforementioned technique division a different split in shape description/analysis techniques is possible. Initially proposed by [Pavilidis78], shapes can be analyzed at the boundary level or interior/global level. Boundary level analysis techniques focus on the contour points of an object, while global level techniques capture global information of a given shape by analyzing not only the boundary points but the interior shape content as well. By combining the shape analysis categorizations (scalar description vs. non-scalar representation and boundary vs. global) most of the current shape description and analysis techniques fall into the following four classes:

Boundary Scalar Techniques: Describe the boundary of a shape by numerically encoding the contour of shape.

Boundary Space-Domain Techniques: Also describe the contour of a shape but use non-numeric descriptors.

Global Scalar Techniques: Use global shape properties, encoded numerically, to describe the given shape.

Global Space-Domain Techniques: Use non-numeric description of the global shape properties to encode a shape description.

The rest of this section provides a brief summary of the most popular and effective methods within the boundary representation class of shape descriptors.

2. BOUNDARY SCALAR TECHNIQUES

Boundary Scalar techniques provide a numerical description of a shape's boundary region. The encoding of a shape is performed by creating 1-D function describing the shape's 2-D contour. The 1-D function is then used to describe the 2-D boundary by either a) applying a Fourier Transform to the 1-D descriptor or b) modelling the 1-D *characteristic* function as a stochastic process.

2.1. **1-D characteristic functions.** Several methods of describing the contour of a shape are commonly used. Perhaps the easiest method is the *centroid-to-boundary distance* approach depicted in Figures 1 and 2. The 1-D function encodes the distance r from the centroid of the shape to its boundary with respect to angle θ, $f(\theta) = r$. Several flavors of this method have been used in the past with the main difference being how sample boundary points are selected. For instance boundary points can be selected equidistant from each other or they can be selected such that the central angle between points is a constant value ($\theta_{i+1} - \theta_i = a$). Regardless of technique variations, most of the centroid-to-boundary distance encodings have problems correctly encoding non-convex shapes, as shown in Figure 2.

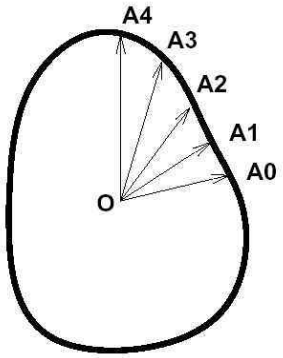

FIGURE 1. Centroid-to-boundary distance approach. [Loncaric98]

Other popular methods of encoding the 2-D shape boundary rely on either computing boundary points *curvature* (representing boundary tangent angular changes) or computing a *complex function* based on either boundary coordinates or the arc length parameters (the approaches are formally outlined in the appendix). Conveniently, all the outlined functions are automatically translation invariant.

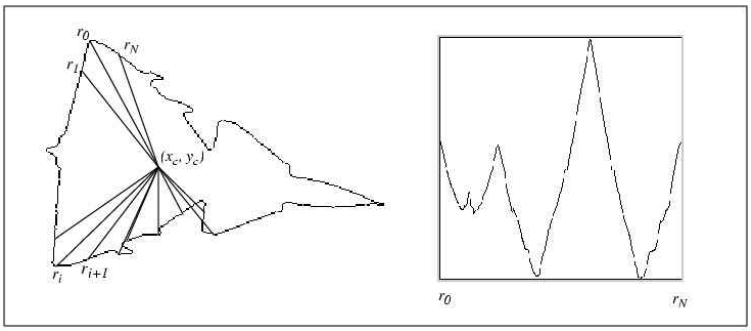

FIGURE 2. Another example of a centroid-to-boundary distance approach with the resulting 1-D characteristic function shown on the right. A limitation of this approach to encoding the boundary can be seen by noticing that several of the rays cross the boundary of the airplane in more than one place. [Kauppinen et al.95]

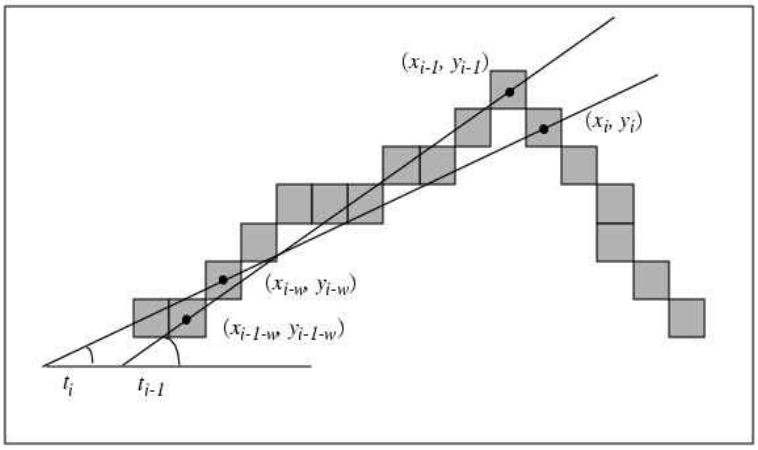

FIGURE 3. Curvature function calculation. $ci = t_i - t_{i-1}$, where t_i and t_{i-1} are two successive tangent values of a boundary segment. [Kauppinen et al.95]

2.2. **Fourier Descriptors.** Once a characteristic function is obtained, a Fourier transform can be used to convert the function from space domain to frequency domain, with the derived sine wave coefficients describing a given 1-D function. While scale invariance is automatically achieved via the transform, a change in rotation angle of the shape results in a phase shift of the sine coefficients. Therefore rotation invariance can also be achieved by looking only at the magnitude of the frequency coefficients. Despite its robustness to scale and rotation variance, the Fourier transform method(s) do not perform well under noisy conditions. To combat noise several solutions have been outlined in [Veltkamp01] and [Kauppinen et al.95]. Because global shape description is encoded by the low frequency components, one successful approach to creating a noise free shape description

is to filter out the noise bearing high frequencies. Another approach is to perform contour smoothing prior to inscribing the shape into the 1-D function as shown in Figure 4. Both methods reduce noise and enable shape generalization but may also merge several distinct shapes together as a side effect.

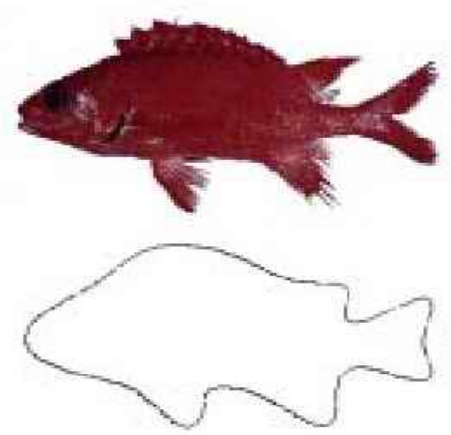

FIGURE 4. Contour smoothing to reduce high curvature changes [Veltkamp01]

2.3. **Stochastic Descriptors.** Methods in this class use the 1-D characteristic shape function as a model of a stochastic process. The goal here is to determine/estimate model parameters which can be used as shape descriptors. Commonly autoregressive techniques (AR) are used in modelling and estimation. A linear autoregressive model estimates the value of a function using m number of preceding values in a linear combination. The general form of the closed[1] AR model [Kashya et al.81] is as follows:

$$r_t = \alpha + \sum_{j=1}^{m} \sigma_j r_{t-j} = \sqrt{\beta}\omega_t \tag{1}$$

where $\{\alpha, \beta, \theta_1, ..., \theta_m\}$ are the model coefficients to be estimated with $\sqrt{\beta}\omega_t$ modelling random noise (residual error) and α being proportional to the mean of the function value. Studies done by [Kartikeyan et al.89] indicate that linear AR models are not always sufficient to encode non-convex shapes and model over fitting may also lead to poor recognition performance. Accuracy may, however, be improved by using higher order models (ie a larger m) or using non-linear stochastic models (eg. quadratic Volterra model). Another remedy to AR model's inabilities to represent complex shapes, generally due to a small number of parameters, is to combine AR model with a hidden Markov model [He et al. 91]. He's and Kundu's approach was to partition the shape boundary into segments and characterize each segment with an AR model parameters to produce a vector sequence for each shape. The final step in the procedure applies an HMM [Rabiner89] to

[1]the AR model is closed since the boundary of a shape is closed making the function periodic

classify the vector sequence. This approach is not only robust with respect to scale, rotation and translation, but also exhibits good performance in the face of noise, occlusion and shape distortion.

2.4. **Fourier based Vs. AR based Methods.** [Kauppinen et al.95] have conducted a comparison study of AR methods pitted against Fourier Transform methods, with both methods using contour, curvature and complex 1-D functions. The experiments tested for sensitivity to noise, scale invariance, rotation invariance, as well as robustness in the presence of perspective distortion around (x, y, z) axis.

Surprisingly, the stochastic based AR methods specifically designed to handle noise were actually outperformed by Fourier based techniques. The authors speculated that the notable superiority of Fourier based methods (on average) was due to the accuracy of low band frequencies at encoding general shape description. In general it was observed that AR models are superior in discriminating objects within a single class due to their abilities to encode local shape characteristics. On the other hand, Fourier descriptors were better at distinguishing objects belonging to different classes due to their ability to accurately encode global shape information.

3. Boundary Space-Domain Techniques

The aim of boundary Space-Domain approaches is to produce a pictoral, graphical or other non-scalar representations of a shape boundary. The most representative of the boundary space-domain methods are chain codes, syntactic techniques, boundary approximations, and scale-space techniques, which are briefly outlined in the following subsections.

3.1. **Chain Codes.** Initially proposed by [Freeman61] the chain code method encodes the shape boundary as a sequence of connected line segments of specified length and direction [Gonzalez et al.92]. The direction of a segment is coded using the schema depicted in figure 5. An example of a boundary encoded by a chain code is shown in figure 6. To remove the fixed distance dependency the code can be generalized (figure 7) to enable contour discontinuities. In general chain codes suffer from several problems, (1) they tend to be quite long and (2) they are very sensitive to local perturbations in the form of noise, distortion and/or imperfect segmentation. Furthermore chain codes are not scale and rotation invariant. However, some of these problems can be alleviated. First off chain code length can be reduced by using a coarser sample grid (ie larger pixels) at the expense of precision of course. In addition the use of a coarser sample grid has been found to compensate for the scale changes as well. To remove rotational dependencies one can use the first difference of the chain code instead of the code itself. This is accomplished by simply taking the difference of the adjacent elements. In figure 6 the original code of $0, 0, 3, 0, 0, 3, 3, 3, 2, 1, 2, 2, ...$ would have a difference of $0, 3, -3, 0, 3, 0, 0, -1, -1, 1, 0, ...$ Finally the novel approach based on Markov Shape Theory (presented in section 4 may have cure all effect for all the problems exhibited by chain code.

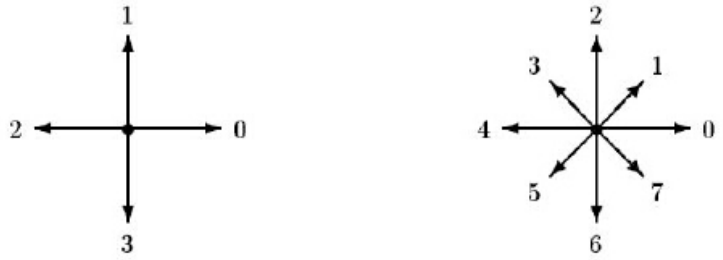

FIGURE 5. Directions for a 4-directional and 8-directional chain code

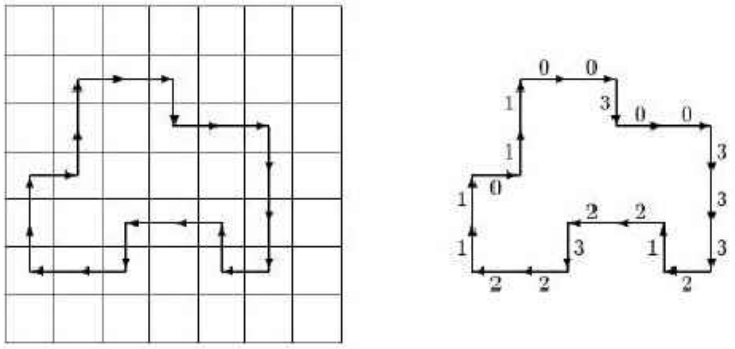

FIGURE 6. An example of a boundary encoded by chain coding. Starting from the top-left boundary point the chain code produced is $0, 0, 3, 0, 0, 3, 3, 3, 2, 1, 2, 2, 3, 2, 2, 1, 1, 0, 1, 1$.

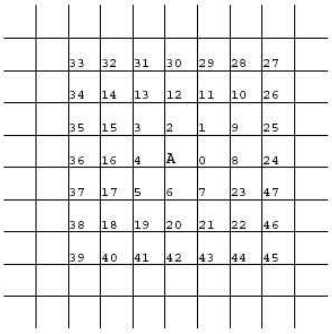

FIGURE 7. Generalized chain code. [Loncaric98]

3.2. **Syntactic Techniques.** Syntactic approach to shape encoding attempts to break the shape boundary into atomic components, which are then subsequently encoded as a string describing the relationship between these constituent components. Formally, the goal of a syntactic techniques then is to create a language that can describe a shape by a *sentence* (string) composed of an *alphabet* of legal symbols (atomic components) using a set of logical rules or *grammar*. Originally formalized by Noam Chomsky, the theory

of formal languages has gained popularity in many fields. As a result one of the main advantages to using syntactic representations is that the field has been extensively developed throughout the years. The main disadvantage is that the shape boundary has to be effectively parsed and encoded by the syntax of the language. This means that the curves, corners and other contour components must all be extracted piece by piece and correctly encoded, a challenge present to the current day.

3.3. **Boundary Approximations.** Typically boundary approximations are accomplished by polygon and spline approximations. The usual procedures within this class of shape descriptors tries to minimize the approximation error while maximizing internal polygon/spline area by using the split and merge approach. The curve segment is split recursively into smaller segments until each segment is approximated within an acceptable error range as shown in figure 8, where a polygon approximation is used. In parallel split segments may also be merged together if the resulting segment does not exceed a maximal error threshold and increases the overall area of the shape approximation.

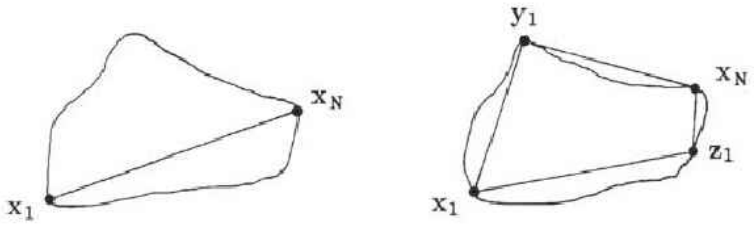

FIGURE 8. A boundary approximation using polygons

3.4. **Scale-Space Techniques.** The underlying principle guiding scale space techniques deals with the notion that points critical the that shapes description are not effected by low-band filtering. Thus, once several Gaussian filters of various widths have been passed over the shape boundary, the remaining contour points are deemed critical to the shape characterization and are stored as shape descriptors.

4. MARKOV SHAPE THEORY

The previous sections have outlined the most common methods for shape description and analysis used in the vision community. Clearly no one method is able to successfully preform shape recognition in the face of noise, distortion, occlusion and various other imperfections present in images and the underlying preprocessing technologies (namely segmentation). The following method is a tentative first look into the use of Markov Shape Theory (MST) to create a robust shape descriptors resistant to the aforementioned image inaccuracies.

The basic premise of Markov Shape Theory deals with the notion that shape information can be directly stored in a Hidden Markov Model. Consider a noiseless boundary of a shape represented by a chain code. Each ”link” in that chain can correspond to a state within a Markov Model and by traversing the chain of states, the state transitions can be

recorded. Thus by using simple frequency counts we can obtain a state transition model from a shape boundary (Eg. the left circle in figure 9). Note that the state transition matrix did not preserve the full shape information. I.e. Shape reconstruction is not possible from just the state transition model. All one is able to do is obtain a similar model from a boundary of the target shape and compare the two models (state transition matrices) for similarities. However, as can be seen in the left image of figure 9, images are not always perfect or complete. Although the reader can make out a circle in the aforementioned figure, the current shape recognition techniques in general cannot effectively handle such broken shape boundaries. In order to successfully recognize a "noisy" or "broken" shape one must rely on both the observations **and** previously learned notion of a circle, which is precisely what the HMM does. Recall that an HMM defined by $\lambda = (\pi, A, B)$ consists of the following [Rabiner89]:

π**:** The probability distribution vector of the starting states.
A**:** The state transition matrix.
B**:** The probability distribution of observations given state.

The A matrix, therefore, stores the model corresponding to a shape, while the B matrix is used as support to link together the model and the observations one acquires from the target image. The overall effect is that a shape is drawn according to both the model and observations. (Note: We still are unable to reconstruct the shape learned by the HMM. A topic saved for the discussion part at the end of the paper)

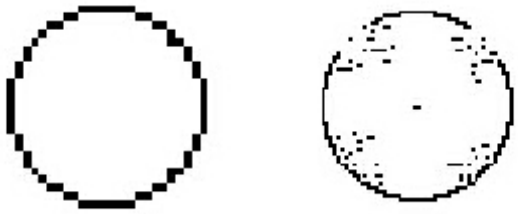

FIGURE 9. Left - a noiseless circle. Right - a circle with noise added to it. Clearly typical boundary encoding algorithms would have problems with the noisy circle due to breaks in the shape boundary

The next sections present the *Viterbi-Max-Max* (VMM) approach based on encoding a shape within an HMM and the experiments conducted using this technique.

4.1. **Viterbi-Max-Max.** The VMM approach is grounded in the boundary domain and uses the chain code to encode state transitions within a closed boundary region. Initially a boundary following algorithm[2] is used to parametrize the contour of a shape in the following way:

$$C(t) = (x(t), y(t)), t \in [0, T] \tag{2}$$

[2]Refer to appendix B.1 for pseudo code of the boundary following algorithm

0.5	0.3	0	0	0	0	0	0.2	0.64103	0.051282	0.10256	0.17949	0.025641
0.28571	0.28571	0.42857	0	0	0	0	0	0.13333	0.23333	0.066667	0.53333	0.033333
0	0.2	0.5	0.3	0	0	0	0	0.051282	0.64103	0.17949	0.10256	0.025641
0	0	0.28571	0.28571	0.42857	0	0	0	0.23333	0.13333	0.53333	0.066667	0.033333
0	0	0	0.2	0.5	0.3	0	0	0.64103	0.051282	0.10256	0.17949	0.025641
0	0	0	0	0.28571	0.28571	0.42857	0	0.13333	0.23333	0.066667	0.53333	0.033333
0	0	0	0	0	0.2	0.5	0.3	0.051282	0.64103	0.17949	0.10256	0.025641
0.42857	0	0	0	0	0	0.28571	0.28571	0.23333	0.13333	0.53333	0.066667	0.033333

FIGURE 10. The A matrix (left) obtained by simple frequency counts and B matrix (right) obtained by a sliding window method. Both matricies obtained from training on the right circle in figure 9

where t can be thought of as a temporal parameter. The contour function starts at an arbitrary position on the (closed) contour and traverses the boundary from start to finish. At each step ($t = i$) the (x, y) coordinates are recorded creating the contour function described by Eq. 2. 8-directional Chain code is then used to encode a function of coordinate changes. The A matrix of an HMM is then populated (and normalized) using frequency counts for state transitions. After training on the chain code obtained from a *clean* circle shown in figure 9 the resulting A matrix can be seen on the left side of figure 10. To train the B matrix the chain code is converted into a series of observations consisting of $-, |, \backslash, /, \emptyset, *$, where $-$ is a horizontal state transition or horizontal edge, $|$ is a vertical edge, $\backslash, /$ are oblique edges, $\emptyset$ indicates no edge present and finally $*$ indicating and edge without a specific direction (Note: last two observations will not happen in a closed boundary, but may occur in test phase on noisy shapes) . A sliding window approach is then used to populate the B matrix. The sliding window approach effectively diffuses the probability distribution curve, thereby allowing for error in the observations (I.e. an observation of a vertical edge may still result in a horizontal state translation given a strong enough model probability of a horizontal state transition). The π vector, plays a small role in our implementation of VMM and can be obtained in a variety of ways, the simplest one being a uniform distribution of starting states. Frequency counts may also be used in determining π as well as the first eigenvector of the A matrix will produce valid results.

Once training has been completed the HMM is used in conjunction with lookahead within the VMM to determine the most probable state transition at each state of the shape traversal. The lookahead uses a *Max-Max* approach (vaguely) represented in figure 11.

At first ply all possible state transitions are considered (in case of figure 11, 4 possible directions may be chosen) in subsequent plies only the maximally likely state transition supported by the observation is chosen. Thus the search greedily selects the best state to expand per trail, given the observation at that position. Formally we use

$$\max_j a_{ij} b_{jk} \tag{3}$$

to select the maximally likely state transition for *each* trail at each ply. Once maximum search depth is reached the path with the highest probability is chosen and the first envisioned state transition is realized (ie. we take a step according to the trail sequence,

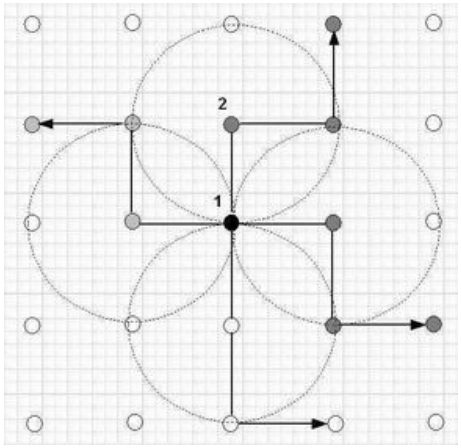

FIGURE 11. Viterbi Max-Max trellis expansion using a 4-directional state transitions. A 4-ply deep expansion is done starting at state 1, at which point the trail with the highest MAP value is chosen. The VMM algorithm then takes one step in the chosen direction state 2 in this case. The procedure then repeats with state 2 as the root node.

in case of figure 11 a state transition from state 1 to state 2 is performed. The procedure repeats it self again until the contour is closed or the trellis expansion steps outside image boundaries.

Once the VMM has stopped, the contour traced by the procedure can be compared to the model. This can be done is several ways. One way is to simply compute the probability of state transitions given observations. If this probability is higher than some pre-specified threshold the target shape is deemed matched to model. Another way of evaluating the closeness of the extracted contour to the model contour is to use the Viterbi algorithm on the sequence of observations collected during the contour traversal and obtain the most likely state sequence. One can then compare, using Hamming distance, the most likely sequence to the actual observed/performed sequence of moves to obtain a measure of how close the contour fits the model.

4.2. **Experiments.** Thus far only rudimentary experiments have been performed. The system was trained on the most primitive of shapes and then tested on either noisy versions of target shapes or shapes belonging to a different class all together. Figures 12 - 18 show the various experiments together with the corresponding results. The first section of experiments deals with using a circle as a training shape and passing in a noisy version of a circle. The VMM is then allowed to run at several ply depths (1-8). The results clearly demonstrate several important properties of the VMM algorithm.

- There does not seem to be any single lookahead ply depth consistently successful at shape recognition. As noise conditions are varied different search depths perform differently and as shown in figure 14, the VMM shape detector can fail at every search depth tested. It is of course conceivable that at higher ply depth recognition could have succeeded. However the point is quite valid. VMM is not

a perfect shape detector/contour extractor, but nonetheless the algorithm shows to be robust to noise in most situations tested.

- Continuing on the previous point, not only no single ply depth is appropriate for all situations, but performance does not necessarily increase with deeper lookahead as demonstrated in figure 16.
- VMM has the ability to perform shape discrimination. Figure 17 demonstrates the results of trying to recognize a (noise free) circle by a model trained to recognize squares. In contrast figure 18 shows that a model trained on a circle will respond to a square and, speculatively to any other shape as well.

The experimental results show that although not perfect, the VMM algorithm *is* indeed quite robust to noise and is able to partially perform shape discrimination. More Experiments will, of course by need to determine effectiveness of the VMM algorithm as compared to classical approaches outlined in sections 2 and 3.

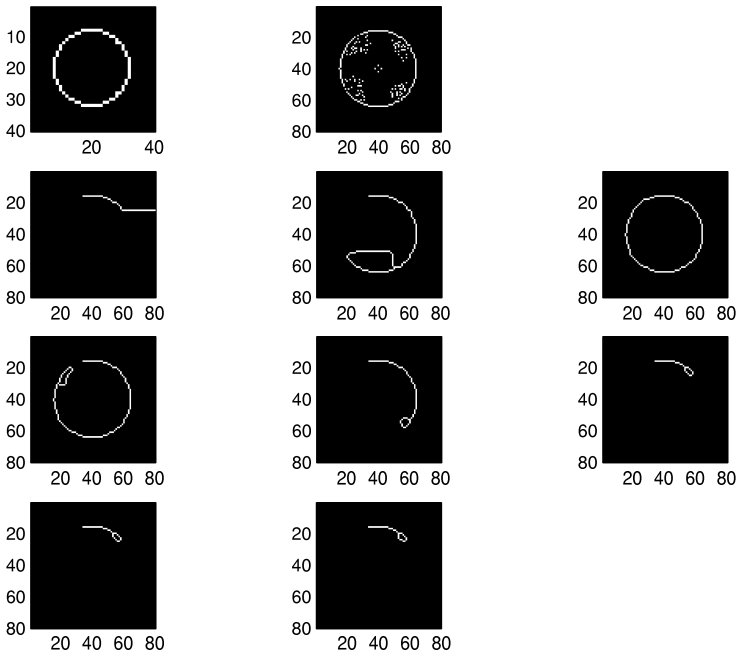

FIGURE 12. Top Left - a noiseless circle used for training. Top Middle - a test circle with noise added to it by using the canny edge detector. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape.

5. DISCUSSION

In the previous section we discussed the inability of HMM at original shape reconstruction. We now present some of the possible solutions to this problem.

5.1. **Model Priming.** A low order Fourier descriptor has been shown to effectively store global shape characteristics. We could draw the course shape outline and then apply the HMM model to fill in the details Figure 16 demostrates a result where a partial square was completed by the VMM. Much like the discussion on AR vs Fourier based methods, where it was shown that the AR models are more effective at storing local shape information effectively whereas the Fourier descriptors are effective at storing global shape properties, so too can the HMM be used to fill in local shape characteristics with the Fourier descriptor providing the *primer* or global information to start the detailed tracing process. Conceptually the Fourier descriptor (or any other coarse boundary descriptor) can be thought of as the catalyst needed to start the recall process that will extract the memory locked with the HMM.

5.2. **Acknowledgements.** Deepest thanks to Dr. Caelli and Dr. Bischoff for their patience and insightful suggestions, without whom this project would not be possible.

5.3. **Conclusion.** This paper presented an overview of current boundary encoding techniques together with a novel approach at model based shape encoding and recognition. While Markov Shape Theory in general and VMM specifically do not provide a cure all method for shape recognition in the presence of noise, occlusion, distortion. These new techniques offer an initial glimpse into possible uses of Hidden Markov Models as robust boundary extraction algorithms.

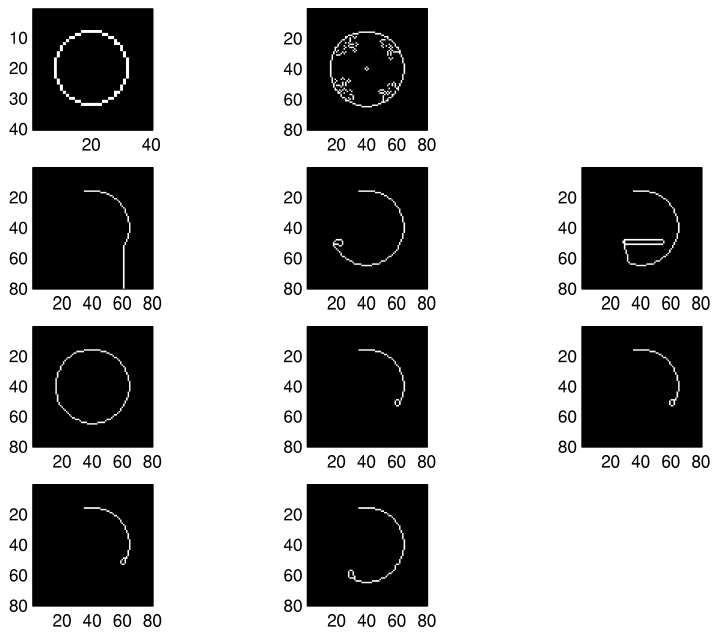

FIGURE 13. Top Left - a noiseless circle used for training. Top Middle - a test circle with noise added to it by using a sobel edge detector. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape.

REFERENCES

[Veltkamp01] R.C.Veltkamp, *Shape Matching: Similarity Measures and Algorithms*, Dept of CS, Utrecht UofPadualaan, 2001.

[Caelli et al.00] T.Caelli, A.McCabe, G.Briscoe, *Shape Tracking and Production using Hidden Markov Models*, International Journal of Pattern Recognition and Artificial Intelligence, 15, 1, 197-221, 2000.

[Pitas00] I.Pitas, *Digital Image Processing algorithms and Applications*, John Wiley & Sons, 2000.

[Loncaric98] S.Loncaric, *A Survey of Shape Analysis Techniques*, Department of Electronic Systems and Information Processing, Faculty of Electrical Engineering and Computing, University of Zagreb, Tech. Report, 1998.

[Tarr et al.98] M.J. Tarr, H.H.Bulthoff, *Image-Based Object Recognition in Man, Monkey, and Machine*, 1998.

[Kauppinen et al.95] H.Kauppinen, T.Seppanan, M.Pietikainen, *An Experimental Comparison of Autoregressive and Fourier-based Descriptors in 2-D Shape Classification*, 1995.

[Gonzalez et al.92] R.C.Gonzalez, R.E.Woods, *Digital Image Processing*, Addison Westley Publishing Company, Inc., 1992.

[He et al. 91] Y.He, A.Kundu, *2-D shape classifcation using hidden Markov model*, IEEE Transactions on PAMI, 13:1172-1184, 1991.

[Rabiner89] L.R.Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, In Proc. IEEE, Vol 77(2), 1989.

[Kartikeyan et al.89] B.Kartikeyan, A.Sarkar, *Shape description by time series*, IEEE Transactions on PAMI, 11:977-984, 1989.

[Aloimonos88] Y. Aloimonos, *Visual shape computation*, Proceedings of the IEEE, 76:899-916, 1988.

[Kashya et al.81] R.Kashyap, R.Chellappa, *Stochastic models for closed boundary analysis: Representation and reconstruction.*, IEEE Transactions on Information Theory, 27:627-637, 1981.

[Hall79] E.L.Hall, *Computer Image Processing and Recognition*, Academic Press, Inc., 1972.

[Pavilidis78] T.Pavlidis, *A review of algorithms for shape analysis*, Computer Graphics and Image Processing, 7:243-258, 1978.

[Marr76] D. Marr, *Early processing of visual information*, Proceedings of the Royal Society of London, B275:483-519, 1976.

[Freeman61] H.Freeman. *On the encoding of arbitrary geometric confgurations*, IRE Transactions, 10:260-268, 1961.

APPENDIX A. MATHEMATICAL FORMULATIONS USED

A.1. **Contour smoothing.** From [Veltkamp01]

If contour C is parameterized by arc-length $s : C(s) = (x(s), y(s))$ then we can perform contour smoothing by convolving the contour C with a Gaussian filter ϕ_σ of width σ as follows:

$$x_\sigma(s) = \int s(x)\phi_\sigma(t-s)dt, \phi_\sigma(t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{t^2}{2\sigma^2}} \tag{4}$$

A.2. **Moments.** From [Loncaric98]

The two-dimensional Cartesian moment $m_{p,q}$ of order $p+q$ for a function $f(x,y)$ is defined as:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy \tag{5}$$

A.3. **Curvature.** From [Kauppinen et al.95]

The curvature of a boundary point i can be computed as follows:

$$c_i = tan^{-1}\frac{y_i - y_{i-w}}{x_i - x_{i-w}} - tan^{-1}\frac{y_{i-1} - y_{i-1-w}}{x_{i-1} - x_{i-1-w}}, i \in [1, N-1] \tag{6}$$

where N is the number of boundary points and w is the window size. Figure 3 shows the approach with a window size of 9 pixels.

A.4. **Complex Coordinate Function.** From [Kauppinen et al.95]

This is simply the coordinates of boundary points w.r.t. objects center.

$$z_i = (x_i - x_c) + j(y_i - y_c) \tag{7}$$

where (x_c, y_c) are the coordinates of the object's centroid and $j = \sqrt{-1}$.

A.5. **Complex Arc Length Function.**

APPENDIX B. VARIOUS PSEUDO CODE

This appendix presents some of the algorithmic details used to create the VMM code.

B.1. **Boundary Following Algorithm.** The following boundary following algorithms was used to extract the boundary of a given shape.

(1) Find starting pixel $s \in S$ for the region using a systematic scan (from left to right, top to bottom).
(2) Let the current pixel in boundary tracking be denoted by c. Set $c = s$ and let the 4-neighbour to the west of s be $b \in \overline{S}$
(3) Let the 8-neighbors of c starting with b in clockwise order be $n_1, n_2, ..., b_8$. Find n_i for the first i that is in S.
(4) Set $c = n_i$ and $b = n_{i-1}$
(5) Repeat steps 3 and 4 until $c = s$.

NOTE: Here 4/8 neighbor refers to the 4/8 directions for a chain code as depicted in Figure 5

E-mail address: ilya@cs.ualberta.ca

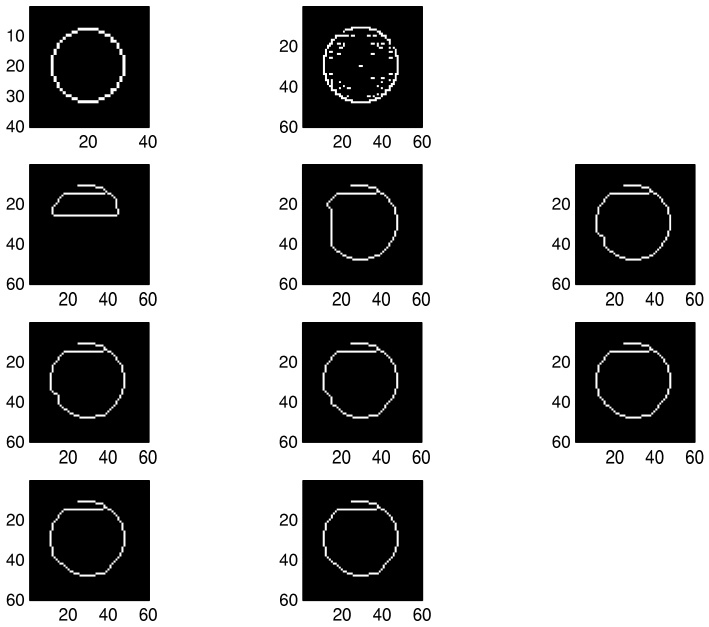

FIGURE 14. Top Left - a noiseless circle used for training. Top Middle - a test circle with noise added to it by using a roberts edge detector. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape.

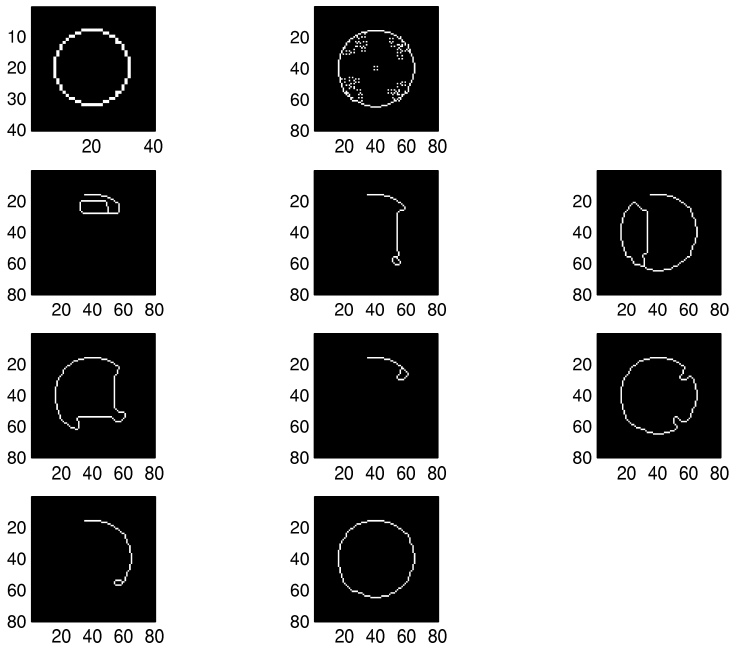

FIGURE 15. Top Left - a noiseless circle used for training. Top Middle - a test circle with noise added to it by using a prewitt edge detector. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape.

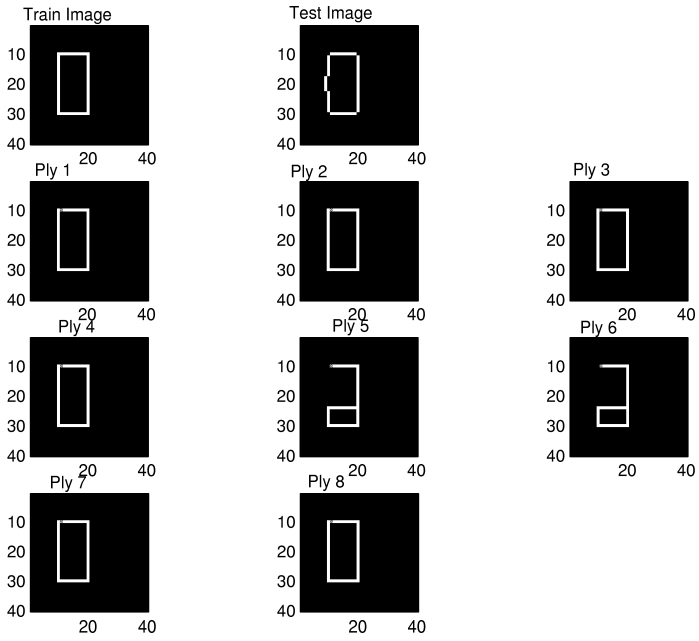

FIGURE 16. Top Left - a noiseless square used for training. Top Middle - a test square with noise added to it. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape. Clearly deeper lookahead is *not* always beneficial as demonstrated by the results obtained using VMM with a ply depth of 5 and 6

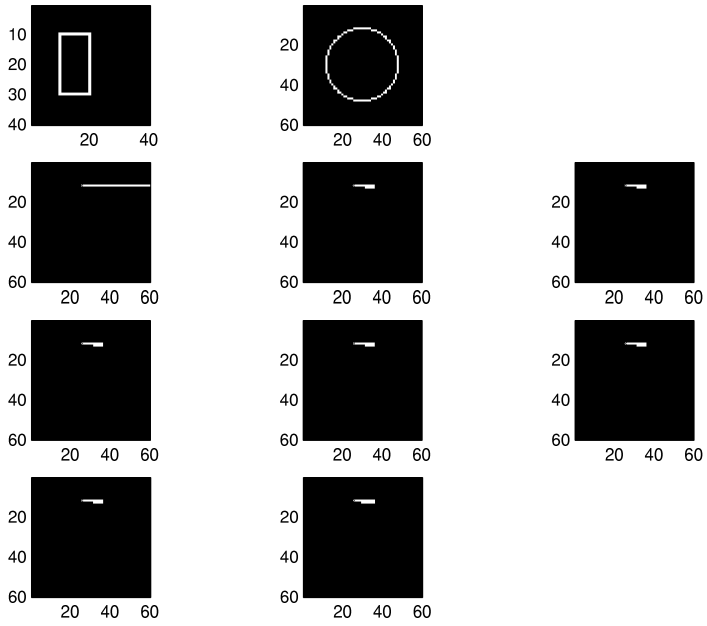

FIGURE 17. Top Left - a noiseless square used for training. Top Middle - a test circle. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 and 8 used to track the test shape. As can be seen a model trained on a square shape does not respond to a circular test shape.

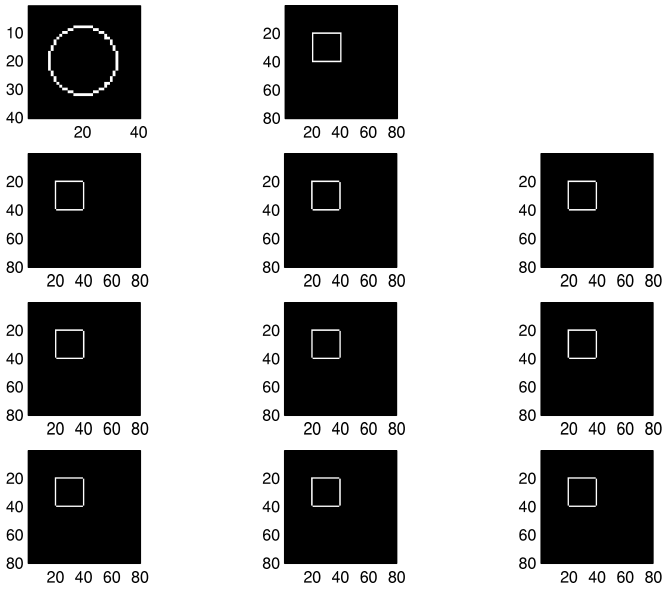

FIGURE 18. Top Left - a noiseless circle used for training. Top Middle - a test square. Results obtained using the VMM tracker. Second Row Ply depth 1-3, third Row ply depth 4-6, bottom row ply depth 7 - 9 used to track the test shape. As can be seen a model trained on a circle shape respond to a square test shape due the fact that a circle is the most general shape. Speculatively, an HMM trained on a circle should be able to represent any given shape.