A Bayesian Multiple Hypothesis Approach to Contour Grouping

Ingemar J. Cox[1] *and James M. Rehg*[2] *and Sunita Hingorani*[1]

[1] NEC Research Institute, 4 Independence Way, Princeton, NJ 08540.
[2] Dept. of Electrical and Computer Eng., Carnegie Mellon University, Pittsburgh, PA

Abstract.

We present an approach to contour grouping based on classical tracking techniques. Edge points are segmented into smooth curves so as to minimize a recursively updated Bayesian probability measure. The resulting algorithm employs local smoothness constraints and a statistical description of edge detection, and can accurately handle corners, bifurcations, and curve intersections. Experimental results demonstrate good performance.

1 Introduction

The set of image contours produced by objects in a scene encode important information about their shape, position, and orientation. Image contours arise from discontinuities in the underlying intensity pattern, due to the interaction of surface geometry and illumination. A large body of work, from such areas as model-based object recognition [8] and contour motion flow [7], depend critically on the reliable extraction of image contours.

The contour grouping problem[1, 10, 6, 12] involves assigning edge pixels produced by an edge detector [4, 3] to a set of continuous curves. Associating edge points with contours is difficult because the input data (from edge detectors) is noisy; there is uncertainty in the position of the edge, there may be false and/or missing points, and contours may intersect and interfere with one another. There are four basic requirements for a successful contour segmentation algorithm. First, there must be a mechanism for integrating information in the neighborhood of an edgel to avoid making irrevocable grouping decisions based on insufficient data. Second, there must be a prior model for the smoothness of the curve to base grouping decisions on. This model must have an intuitive parameterization and sufficient generality to describe arbitrary curves of interest. Third, it must incorporate noise models for the edge detector, to optimally incorporate noisy measurements, and detect and remove spurious edges. And finally, since intersecting curves are common, the algorithm must be able to handle these as well. We believe our algorithm is the first unified framework to incorporate these four requirements.

We formulate contour grouping as a Bayesian multiple hypothesis "tracking" problem. Our work is based on an algorithm originally developed by Reid [11] in the context of military and surveillance tracking of multiple targets (aircraft) in the presence of noise. The algorithm has three main components: A "dynamic" contour model that encodes the smoothness prior, a measurement model that incorporates edge detector noise characteristics, and a Bayesian hypothesis tree that encodes the likelihood of each possible edge assignment and permits multiple hypotheses to develop in parallel until sufficient information is available to make a decision.

Section (2) develops the Bayesian hypothesis tree, and associated probability calculations. Section (3) describes some important implementation details, and is followed by a presentation of experimental results in Section (4). We conclude in Section (5) with a discussion of directions for future work.

2 Multiple Hypothesis Framework for Contour Grouping

In this section we describe the Bayesian multiple hypothesis algorithm [11] and demonstrate its application to the extraction of contours from an edge image. Figure (1) illustrates the principal conceptual components of the contour grouping algorithm. At each iteration, there are a set of hypotheses (initially null), each one representing a different interpretation of the edge points. Each hypothesis is a collection of contours and at each iteration each contour predicts the location of the next edgel as the algorithm follows the contour in unit increments of arc length. An adaptive search region is created about each of these predicted locations as shown in Figure (2) and detailed in Section (3). Measuresurements are extracted from these surveillance regions and matched to predictions based on the statistical Mahalanobis distance. This matching process reveals ambiguities in the assignment of measurements to contours. As a result, the hypothesis tree grow another level in depth, a parent hypothesis generating a series of hypotheses each being a possible interpretation of the measurements. The probability of each new hypothesis is calculated based on assumptions described later. Finally, a pruning stage is invoked to constrain the exponentially growing hypothesis tree. This completes one iteration of the algorithm.

With this overview in mind, a detailed description of the algorithm is now described, following [2, 9]. At iteration k we have a set of association hypotheses Ω^k obtained from the set of hypotheses Ω^{k-1} at iteration $k-1$ and the latest set of measurements, $Z(k) = \{\mathbf{z}_i(k)\}_{i=1}^{m_k}$, where m_k is the number of measurements $\mathbf{z}_i(k)$ at measurement interval k. The measurement set $Z(k)$ is obtained by establishing a surveillance or search region around the predicted end point of each hypothesized contour. The precise description of this surveillance region is postponed until Section (3).

An association hypothesis, $Z^{k,l}$, groups together edgels originating from a single contour. We define $Z^{k,l} \triangleq \{\mathbf{z}_{i_1,l}(1), \mathbf{z}_{i_2,l}(2), ..., \mathbf{z}_{i_k,l}(k)\}$ to be the set of all measurements originating from contour l. An association hypothesis is one possible interpretation of the current edgels. Due to ambiguity in interpreting edgels, there will be many association hypotheses. The resulting partitions are disjoint, however, since an edgel can only belong to a single contour. By means of a Bayesian multiple hypothesis tree, each hypothesis can be assigned a probability value through a recursive formula.

New hypotheses are formed by associating each measurement as either belonging to a previously known contour, or being the start of a new contour or false alarm. In addition, for contours that are not assigned measurements, there is the possibility of termination of a contour. We define a particular global hypothesis at iteration k by Θ_m^k. Let $\Theta_{l(m)}^{k-1}$ denote the parent hypothesis from which Θ_m^k is derived, and $\theta_m(k)$ denote the event that indicates the specific status of all contours postulated by $\Theta_{l(m)}^{k-1}$ at iteration k and the specific origin of all measurements received at iteration k. In general, $\theta_m(k)$ will consist of τ measurements from known contours, ν measurements from new contours, ϕ false alarms and χ terminated (or ended) contours from the parent hypothesis. Figure (2) illustrates a situation in which we have two known contours (T_1 and T_2) and three new measurements $\{\mathbf{z}_1(k), \mathbf{z}_2(k)$ and $\mathbf{z}_3(k)\}$.

We can construct an event $\theta_m(k)$ by creating a *hypothesis matrix* in which known contours are represented by the columns of the matrix and the current measurements by the rows. A non-zero element at matrix position $c_{i,j}$ denotes that measurement $\mathbf{z}_i(k)$ is contained in the validation region of contour t_j. In addition to the T known contours, the hypothesis matrix has appended to it a column 0 denoting false alarms and a column $T+1$ denoting new contour. Hypothesis generation is then performed by picking one unit per row and one unit per column except for columns T_F and T_N, where the number

of false alarms and new targets is not restricted. In this manner we impose the dual constraints that (1) a measurement can originate from only one contour (disjointness) and (2) that a contour produces only one measurement per iteration (this is guaranteed by our adaptive search strategy).

Following [2, 9], we derive a recursive expression for the likelihood of a given segmentation hypothesis conditioned on a set of edge measurements. A given hypothesis at iteration k, Θ_m^k, is composed of a current event and a previous hypothesis resulting from measurements up to and including iteration $k-1$: $\Theta_m^k = \left\{\Theta_{l(m)}^{k-1}, \theta_m(k)\right\}$ We wish to calculate the probability $P\left\{\Theta_m^k | Z^k\right\} = P\left\{\theta_m(k), \Theta_{l(m)}^{k-1} | Z(k), Z^{k-1}\right\}$. Our derivation, presented in detail in [5], is based on two assumptions. First, the numbers of false alarms and new edgels are Poisson distributed with densities λ_F and λ_N, respectively, while contour initiations are distributed uniformly. Second, measurements assigned to a given contour are Gaussian distributed, while false edge measurements are uniformly distributed over the surveillance volume. Under these conditions, we obtain [2]:

$$P\{\Theta_m^k | Z^k\} = \frac{1}{c}\frac{\phi!\nu!}{m_k}\mu_F(\phi)\mu_N(\nu)V^{-\phi-\nu}\prod_{i=1}^{m_k}[N_{t_i}[\mathbf{z}_i(k)]]^{r_i}$$

$$\left\{\prod_t (P_D^t)^{\delta_t}(1-P_D^t)^{1-\delta_t}\left(P_\chi^t\right)^{\chi_t}\left(1-P_\chi^t\right)^{1-\chi_t}\right\} P\{\Theta_{l(m)}^{k-1} | Z^{k-1}\}$$

3 Implementation

In this section, we briefly describe five important components of the segmentation algorithm: the *structure* and *pruning* of the hypothesis tree, *contour* and *measurement models*, a *termination probability model*, the *surveillance volume*, and a post-processing stage called *contour fusion*.

The hypothesis tree organizes segmentation alternatives and their associated likelihoods. Efficient implementation of this tree is critical to the practical success of the algorithm. A single hypothesized contour may be present in many global hypotheses. Rather than replicate this contour for each hypothesis with the associated memory and computational overheads, Kurien [9] proposed the construction of a contour (or track) tree, in which the root denotes the creation of a new contour and each branch denotes an alternative measurement assignment. Each node in the global hypothesis tree contains a set of pointers to track trees. Each set represents a different permutation of contour leaf nodes from different contour trees, i.e. the global hypotheses enforce the assumptions of disjoint partitions. The contour tree provides considerable savings, and is discussed in detail in [9].

Pruning is an essential part of any practical contour segmentation algorithm. In our implementation, pruning is based on a combination of the "N-scan-back" algorithm [9] and a simple lower limit probability threshold. The "N-scan-back" algorithm assumes that any ambiguity at iteration k is resolved by iteration $k+N$. Then, if hypothesis Θ_m^k at iteration k has m children, the sum of the probabilities of the the leaf nodes is calculated for each of the m branches. The branch with the highest probabilty is retained and all others are pruned. This gives the tree a particular structure: below the decision node it has a depth of N, while above the node it has degenerated into a simple list of assignments. In our experiments, N is set to 3. While this may seem quite small, previous tracking results [9] suggest that even $N=2$ can provide near optimum solutions. After this procedure, the number of leaf nodes can still be very high. A second phase of pruning removes all nodes whose probability is less than a lower limit, which is currently set to 0.01.

A key step in assigning probabilities to segmentation hypotheses is the computation of the likelihood that a given measurement originated from a certain contour. This likelihood computation depends on two things: a dynamic model that describes the evolution of the curve in the image, and a measurement model that describes how curves produce edgels. In our formulation, the curve state vector is $[\dot{x}\, x\, \dot{y}\, y]'$ and its dynamics are described by a linear noise-driven acceleration model common in the tracking literature ([2], Chapter 2). The autocorrelation of the white Gaussian acceleration noise can be varied to model curves of arbitrary smoothness. Thus the tip of the contour as a function of arc length, t, is $(x(t), y(t))$ and has tangent $(\dot{x}(t), \dot{y}(t))$. Since many edge detectors provide gradient information, we assume that the entire state vector is available for measurement. A Kalman filter is then employed to estimate curve state and predict the location of edgels. These predictions are combined with actual measurements to produce likelihoods.

Once the location of a given curve has been predicted by the Kalman filter and discretized to image coordinates, a surveillance region is employed to extract measurements. A surveillance region is a set of concentric circles, of radius 1, $\sqrt{2}$, 2, $\sqrt{5}$, that are searched for edgels (the radii define discrete pixel neighborhoods.) It is these measurments that form segmentation hypotheses whose probabilities are computed as described earlier.

The probability of termination controls the distance over which a contour will be extended if no measurements are present. One may want this gap size to be 10's of pixels. However, if N-scan is set to 3 the tree is always pruned such that the contour continues, i.e. the contour is never terminated. Making N-scan substantially increases the size of the tree. Instead, the probability of termination, P_χ is set to $P_\chi = 1 - \exp(-m/\lambda_\chi)$ where λ_χ is a parameter set by the user and m is specific to each contour and is the number of consecutive iterations in which no measurement was assigned to the contour.

Since the algorithm scans the edge image by "walking" along contours, it may encounter a new contour at any point along its length. When tracking begins in the interior of a curve, it is usually partitioned, erroneously, into two or more segments sharing common boundary points. These multiple contours can be merged to recover the correct segmentation, compensating for the incorrect initial conditions. The Mahalanobis distance provides a simple contour merge test: Two contours with state estimates $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$ at a common boundary are merged if $\mathbf{dx}'_{ij}\mathbf{T}_{ij}^{-1}\mathbf{dx}_{ij} \leq \delta$, where $\mathbf{dx}_{ij} = \hat{\mathbf{x}}^i - \hat{\mathbf{x}}_j$ and $\mathbf{T}_{ij}$ is its covariance, and δ is obtained from χ^2 tables. This test is applied after the algorithm produces an initial segmentation.

4 Experimental Results

It is very difficult to quantify the performance of a contour grouping algorithm because the definition of a "correct" segmentation is application-dependent. The experiments presented here are therefore qualitative, illustrating the performance and limitations of the algorithm. For information on parameter settings, the reader is directed to [5].

Figure (3) shows the contour groupings for various images of a fork, a computer mouse and a reasonably complex cutting tool. The ends of the contours are denoted by circles. For the fork, the curvature at the end of the handle was smooth enough to allow the contour to continue around, while the curvature at the prongs of the teeth is too great and the contour is broken. Note that corners, bifurcations and intersections are all reasonably partitioned, even though these concepts are *not* explicitly represented in the algorithm. The algorithm also correctly segments circular contours even though the contour model is piecewise linear.

5 Discussion

We presented a grouping algorithm that finds the optimal Bayesian maximum a posteriori partitioning of edges into contours. It is based on two observations: First, grouping

decisions must be based on a prior model for curve smoothness. Second, the difficulty of the segmentation problem can be expressed by sensor/environmental statistics such as mean rates of false edgels and new contours. The resulting algorithm is physically grounded, i.e. *all* free parameters are physical quantities of the sensor and or scene that can be physically measured. We believe the algorithm is the first unified framework to incorporate a measurement noise model, scene statistics, optimal state estimation for the contour model, statistical distance measures to quantify "closeness" and Bayesian decision trees in a recursive formulation that is independent of the image sampling grid.

There are several avenues for future work. First, it would be useful to investigate other contour models, such as a piecewise constant curvature model. It would also be interesting to tightly couple the edge detection and contour grouping stages, with the latter providing the former with expectations of where to look for edges. This unification may improve upon standard techniques for curve enhancement, such as hysteresis [4]. Finally, we would like to extend the Bayesian formulation to incorporate application specific segmentation requirements. In a recognition scenario, for example, partial identification of the scene could modify the prior probabilities associated with the contour grouping algorithm.

The authors would like to thank Y. Bar-Shalom, H. Durrant-Whyte, T. Kurien and J. J. Leonard for valuable discussion on issues related to target tracking.

References

1. D. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.
2. Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association.* Academic Press, 1988.
3. R. A. Boie and I. J. Cox. Two dimensional optimum edge detection using matched and wiener filters for machine vision. In *IEEE First International Conference on Computer Vision*, pages 450–456. IEEE, June 1987.
4. J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):34–43, 1986.
5. I. J. Cox, J. M. Rehg, and S. Hingorani. A bayesian multiple hypothesis approach to contour grouping. Technical report, NEC Research Institute, Princeton, USA, 1991.
6. C. David and S. Zucker. Potentials, valleys, and dynamic global coverings. *Int. J. Computer Vision*, 5(3):219–238, 1990.
7. E. Hildreth. Computation underlying the measurement of visual motion. *Artificial Intelligence*, 27(3):309–355, 1984.
8. D. Kriegman and J. Ponce. On recognizing and positioning 3-d objects from image contours. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(12):1127–1137, December 1990.
9. T. Kurien. Issues in the design of practical multitarget tracking algorithms. In Y. Bar-Shalom, editor, *Multitarget-Multisensor Tracking: Advanced Applications*, pages 43–83. Artech House, 1990.
10. A. Martelli. An application of heuristic search methods to edge and contour detection. *Communications of the ACM*, 19(2):73–83, 1976.
11. D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24(6):843–854, December 1979.
12. A. Shashua and S. Ullman. Grouping contours by iterated pairing network. In Richard P. Lippmann, John Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems 3.* Morgan Kaufmann, 1991. Proc. NIPS'90, Denver CO.

This article was processed using the LaTeX macro package with ECCV92 style

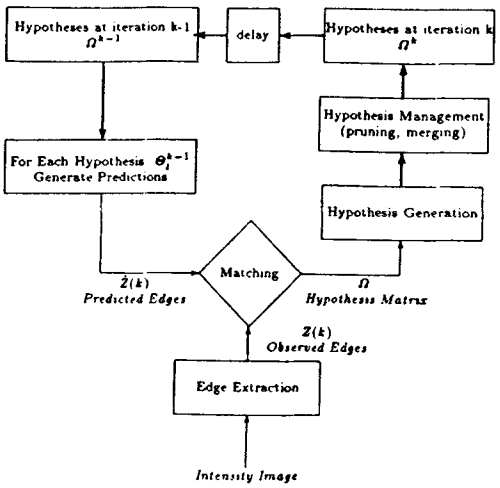


Fig.1. Outline of the contour grouping algorithm.

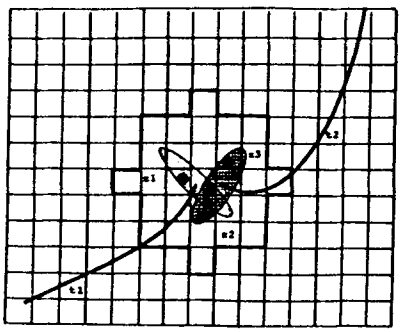
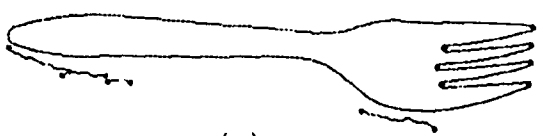
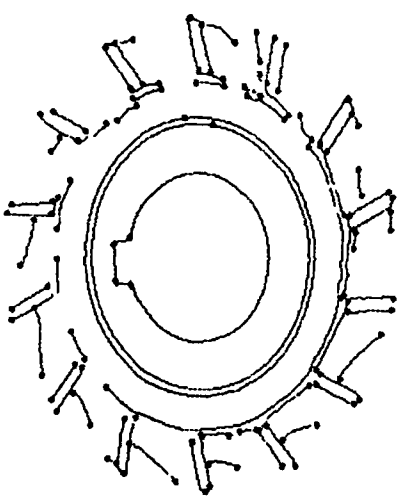


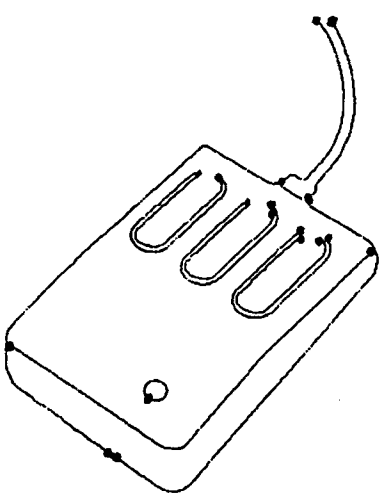
Fig.2. Predicted contour locations, a surveillance region and statistical Mahalanobis (elliptical) regions.



(a)



(c)



(b)

Fig.3. Contour groupings for (a) fork, (b) mouse and (c) cutter. Circles denote contour end points.