Color and Shape Based Tracking

Victor Z. Wu Evangelos E. Milios
Department of Computer Science
York University
North York, Ontario, Canada, M3J 1P3
g-wu@cs.yorku.ca eem@cs.yorku.ca

Abstract

In this paper we address the problem of tracking objects based on their color and shape. The main idea is to update the color model of the object as lighting conditions and projection geometry change with the movement of either the camera or the object.

As the color model of the object being tracked we use the color histogram obtained from all the pixels which are interior to its closed boundary. We backproject the image at time t+1 through the color histogram obtained from the object in the image at time t. The result is a greyscale image, the backprojected image. The backprojected image is bright where the object is and dark elsewhere. To update the boundary of the object we apply a snake model to the backprojected image. Snakes are attracted by step changes in image intensity. We demonstrate that the snake is capable of converging to the correct object boundary at time t+1 in most cases. Minor convergence problems of the snake can be easily addressed by employing color histogram matching over appropriate image regions.

An X-window based interactive program has been implemented, that allows users to display streams of images, define a model on an image, draw snakes, and execute other image and snake operations through a menu-based interface.

1 Introduction

The problem which we are addressing is the following: A mobile robot is equipped with a CCD color camera. The goal is to enable the CCD camera to track a color object (**target**). Both the robot and the target may be either stationary or moving. By "tracking" we mean knowledge of *the position of the target's projection in the image plane* (defined by the contour of the projection) *at all times.* If the camera is on a pan and tilt platform [3,4], tracking allows the camera to keep the target within its view all the time.

In this paper, we mainly address the following task: given a stream of images and the location of the target's projection in the first image, search for the occurrences and compute the locations of the target in the rest of the images.

The engineering problems that we are facing here are two-fold. First is the need for automation. We need to build up a system which searches for the target itself, if possible with little human assistance in the beginning. After all, the purpose of the tracking is to add "vision" to an autonomous robot. Secondly, there is a need to develop "qualitatively accurate" as well as "numerically efficient" techniques. Although we aim specifically at the tracking problem, we would also like the techniques developed here to be applicable to a number of essential computer vision problems. By "qualitatively accurate" we mean that the techniques can supply meaningful qualitative information, such as the location and contour of the shape. Furthermore, such information should be robustly computed in the presence of noise. Numerical efficiency refers to providing implementations which are affordable by normal computing resources and time constraints.

2 Color in Recognition and Tracking

Given a color space defined by some color axes (e.g. red, green, blue), **the color histogram of a color image** is obtained by counting the number of times each color occurs in the image [8]. To see this, assume that the color space is divided into cubic buckets (Figure 1), each bucket corresponding to a unique set of colors. In a color histogram, each bucket stores a number which corresponds to the number of pixels in the image which have the colors belonging to the set of colors.

We compare objects by judging the similarity of their color histograms.The similarity of two color histograms may be illustrated by their intersection and match value [8] as described below.

Given a pair of histograms, I and M, each containing n buckets $I_1, \ldots, I_n$ and $M_1, \ldots, M_n$, with I_i and M_i represent the same set of colors for $i = 1, \ldots, n$, the **intersection of I and M** is defined as:

$$\sum_{i=1}^{n} \min(I_i, M_i) \tag{1}$$

and the **match value of I with respect to M** is defined as:

$$match(I, M) = \frac{\sum_{i=1}^{n} \min(I_i, M_i)}{\sum_{i=1}^{n} M_i} \tag{2}$$

We use the match value to determine if two objects are the same, i.e., object O_i (with histogram

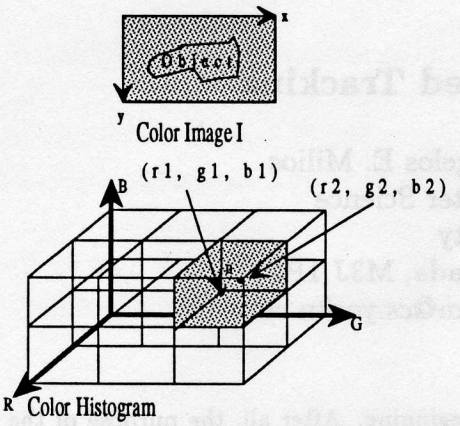

Figure 1: The shaded bucket in the histogram represents the set of colors $C = (r, g, b)$ such that $r1 \leq r < r2$, $g1 \leq g < g2$, and $b1 \leq b < b2$. n is the number of pixels in the color image I of colors belonging to the set C.

I) is identical to object O_m (with histogram M) if $match(I, M) > \delta$, where δ is a threshold.

How reliable is the above criterion for identifying color objects? We can think of each color histogram M as a point in a multi-dimensional metric space, E, and hence the above criterion defines a metric sphere centered at M with radius δ. Swain [8] proved that the fraction of the multi-dimensional metric space E occupied by a single model is at most

$$\frac{(2(1-\delta))^{n-1}}{\sqrt{n}}$$

where n is the number of the buckets (dimension of the metric space), and δ is the threshold as in eq.2. For example, if we choose $\delta = 0.6$ and $n = 512$, the fraction is about 10^{-51}. Hence, if the histograms were distributed evenly throughout the color space, the probability of the failure of the criterion is approximately 10^{-51}. Even though histograms may not distribute evenly throughout histogram space, this criterion still works well by reducing the number of buckets in histograms (*blurring*).

But the above criterion will **not** work alone to solve the tracking problem. As a counterexample, suppose that the target only occupies 10% of the whole image, then even though it moves out of the scene, the two images with and without the target still match according to the criterion.

Thus, we have to find a way to narrow down the area in the image where the target will most probably occur. We do that by first transforming the color images into conventional intensity images as described in the following section.

3 Histogram Backprojection

The concept of Color histogram backprojection[8] is depicted in Figure 2.

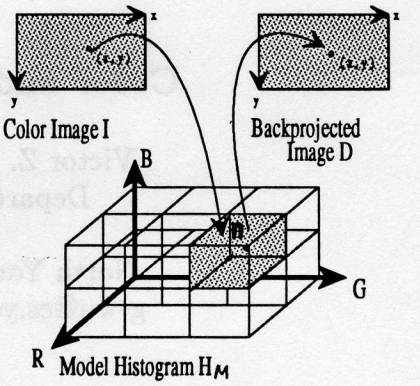

Figure 2: If pixel (x, y) in I has color which belongs to the set of colors represented by the shaded bucket in the histogram H_M, then the pixel (x, y) in D has intensity value n which is the number stored in the shaded bucket in H_M.

What makes the histogram backprojection useful? As you can see from the illustration, provided colors of the target are fairly dense, and different from the colors of other objects, the backprojected image (Image Plate (a), (b)) will basically be black except the area(s) corresponding to the instance(s) of the model, where blob(s) (regions of high intensity responses) will be formed. The shape of the blob will be similar to the shape of the target except where the target is partially.

In practice, we first compute the hue and saturation values, h and s, from r, g, and b using the well-known transformation equations from RGB to IHS space [5], and we construct the histogram using only the hue and the saturation axes in the IHS space [2]. Then, we compute the backprojected image accordingly.

4 The Active Contour Model

We use **snakes** [1] to find the location (contour) of the blob corresponding to the target in the backprojected image, given the approximate position of the blob. More specifically, let the snake be represented by

$v(s) = (x(s), y(s))$, s is in $[a, b]$, and $v(a) = v(b)$

The snake's energy functional is defined as

$$E^*_{snake} = \int_a^b E_{snake}(v(s))ds \quad (3)$$

$$\equiv \int_a^b [E_{int}(v(s)) + E_{image}(v(s))]ds \quad (4)$$

where E_{int} is the internal spline energy which imposes a smoothness constraint on the shape of the snake. E_{image} is the image energy which could be used to attract the snake to (or push the snake away from) the salient image features like lines, edges etc.

Our goal is to find $v^*(s)$ such that

$$E^*_{snake}(v^*(s)) \equiv \min_{v(s)} E^*_{snake}(v(s)) \quad (5)$$

We will show that by properly defining E_{snake}, it is possible to compute v* which is also the boundary of the desired region (the blob) which we are looking for.

4.1 The Internal Energy

The internal spline energy is defined as

$$E_{int} \equiv \frac{\alpha(s)|v'(s)|^2 + \beta(s)|v''(s)|^2}{2} \quad (6)$$

The first order (derivative) term, $\alpha(s)|v'(s)|^2$, makes the snake behave like a **string** or **membrane** (i.e. resists stretching), while the second order term, $\beta(s)|v''(s)|^2$, makes the snake act like a **rod** or **thin plate** (i.e. resist bending). Setting $\beta(s)$ to zero at a point allows a tangent discontinuity at that point and develops a corner. Setting both $\alpha(s)$ and $\beta(s)$ to zero removes the internal constraint completely at that point and allows a position discontinuity.

4.2 The Image Energy

The purpose of the image energy functional is to make the snake be attracted to the salient image features. We are interested in finding the contour of a high intensity blob in the backprojected image, hence we can define the image energy functional as

$$E_{image} \equiv \omega_{high} I(x,y) + \omega_{edge} |\bigtriangledown I(x,y)|^2 \quad (7)$$

where I(x,y) is the intensity of pixel (x,y) in the backprojected image, $|\bigtriangledown I(x,y)|^2$ is the image gradient at point (x,y), and ω_{high}, ω_{edge} are weight factors.

Once again, the weights here can affect the behavior of the snake. For example, setting the sign of ω_{high} and ω_{edge} to negative makes the snake be attracted to high intensities and high image gradient pixels (points on the contour) while setting their sign to positive pushes the snake away from high intensities and high image gradient points.

4.3 Computation of Snakes

A method of computing $v^*(s)$ as defined in 5 has been discussed in detail in [1,9]. v^* is computed as follows: Starting with an initial estimation of v^*, say v_0, we compute the next approximation of v^*, say v_1, using v_0, ... The sequence v_0, v_1, ... normally will converge to v^*.

It turns out that each step of the above computation has only O(n) complexity where n the number of snake points (mesh points of $[a,b]$). On our SPARC II workstations with 64MB memory and 21MIPS speed under average day-time workload, our C procedure of the above computation takes less than two seconds for 1000 iterations.

5 The Tracking System

We present here the algorithms and our interactive interface called COTS (Color Object Tracking System). Experiments and their results are reported and discussed as well.

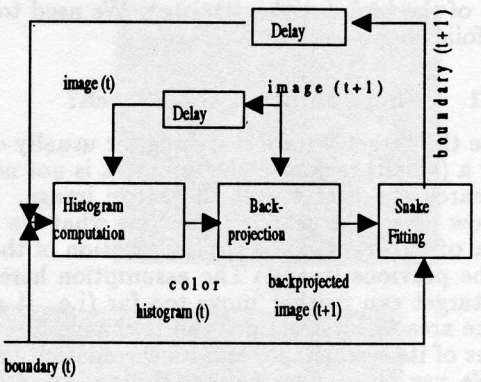

Figure 3: The top-level logic structure of COTS.

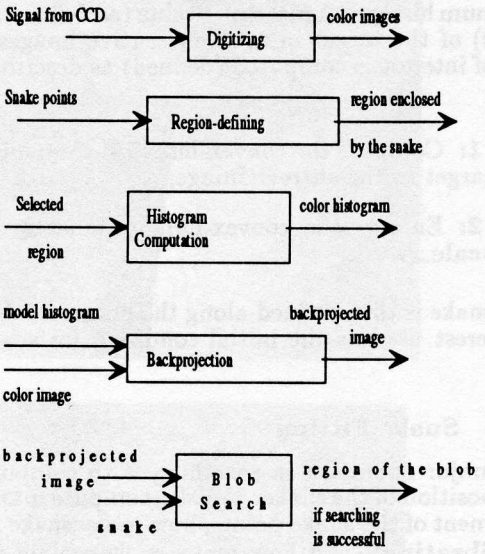

Figure 4: Top-level modules for COTS

5.1 Top-Level Modules

The top-level logic structure of COTS is depicted in Figure 3. We compute the color histogram at time (t) of the region bounded by the computed boundary at time (t) in the color image at time (t). (Initially, the boundary of the target is defined interactively.) We then use the computed histogram and the next image frame at time (t+1) to compute the backprojected image at time (t+1). We then search for the blob in the backprojected image and the contour of the blob is used as the boundary at time (t+1). The color histogram is then updated (re-computed) using boundary at time (t+1) and image at time (t+1). This completes one iteration of the loop.

The top-level modules of the COTS is depicted in Figure 4.

5.2 The Searching Algorithm

How do we search for blobs? The idea here is to carry out the search by fitting a snake along the con-

tour of the blob (**snake fitting**). We need to answer the following questions:

5.2.1 Where to Start the Search?

Since the target we are searching for usually occupies only a (small) region in the image, it is not necessary to search the target over the entire image. We can narrow down the search region, or what we call the **area of interest**, knowing the location of the target in the previous image. The assumption here is that the target can neither move too far (i.e. it must be in the area of interest) nor can it change too much in terms of its size, shape, and its chromaticity.

We use three parameters, **d_size**, **match_value** and **scale_ratio**, to quantify the above concepts. d_size is the maximum change in size allowed for the target in two consecutive images. match_value is the minimum histogram matching value (as defined in section 2) of the target in two consecutive images. The area of interest is computed (defined) as described below:

Step 1: Compute the convex-hull [6,7] containing the target in the current image;

Step 2: Enlarge the convex-hull by a factor called **scale_ratio**.

A snake is then defined along the border of the area of interest used as the initial condition for snake fitting.

5.2.2 Snake Fitting

The major operation in searching is to compute the next position of the snake. We also compute maximum movement of the snake points, how many snake points are **vibrating**[1], and how many snake points are on fairly high intensity spots in the backprojected image. These three values are then used to decide what to do next, as described below:

1. if the movement is too radical, it most probably means the step size γ [1] is too large. Hence we decrease the step size γ by a fixed factor and run another iteration on the old snake points;
2. if most of the snake points (e.g. 75% of them) are vibrating, this means those vibrating points are near some local edges. Hence it is a good time to check if we have found the target by computing the match value of the histogram of the area enclosed by the snake and the model histogram. If the match value is big enough and the size difference of the area and the model is smaller than d_size, the search succeeds. If the new area is too large, more iterations are performed. If the match value is too small, the search fails.
3. if most of the snake points are on fairly high intensity spots or the snake moves too slowly, do the same thing as in the previous case.

[1] Let (x_t, y_t), (x_{t+1}, y_{t+1}) and (x_{t+2}, y_{t+2}) be the computed locations of the same snake point p in consecutive iterations. We say p is vibrating if $(x_{t+1}-x_t)(x_{t+2}-x_{t+1}) < 0$ or $(y_{t+1}-y_t)(y_{t+2}-y_{t+1}) < 0$.

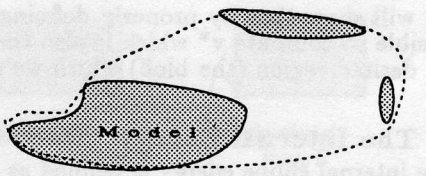

Figure 5: The snake (dotted line) has difficulty to move over the spurious areas depicted by the small shaded areas

In some cases, the snake may have difficulty overcoming the resistance caused by spurious areas (noise) as depicted in Figure 5. This may cause the snake to stop shrinking before it finds the contour of the target (i.e., the new area enclosed by the snake is too large). We solve this problem by **splitting** the region enclosed by the snake into two subregions so that the blob corresponding to the target is in one region and some of the snake-obstructing noise in the other region. Then we discard the unwanted region and redefine the snake along the border of the region chosen, and then do other iterations.

5.2.3 When to Stop

Since we are using finite differences to approximate the differential equations, the common numerical behavior, if the step size is too large, is that the solution vibrates around the equilibrium instead of converging to it. However, if the step size is too small, the convergence will be too slow. We handle this tradeoff by choosing a fairly large step size to speed up the convergence and stopping the iteration using the criteria mentioned in case 2 and 3 in the previous section. If the search is successful (and hence the snake is presumably very close to its equilibrium location), we do some more iterations on the snake points [1] with very small step size γ to allow the snake to converge to its equilibrium.

6 Experiments

6.1 Effects of the Resolution of the Color Spaces

One has to decide how finely to discretize the color space before computing color histograms. We use the term **resolution of the color space** to express the "fineness" of such discretization. The finer the color space is discretized (i.e., the smaller each bucket is), the higher resolution it has, and more space is needed to store the histograms.

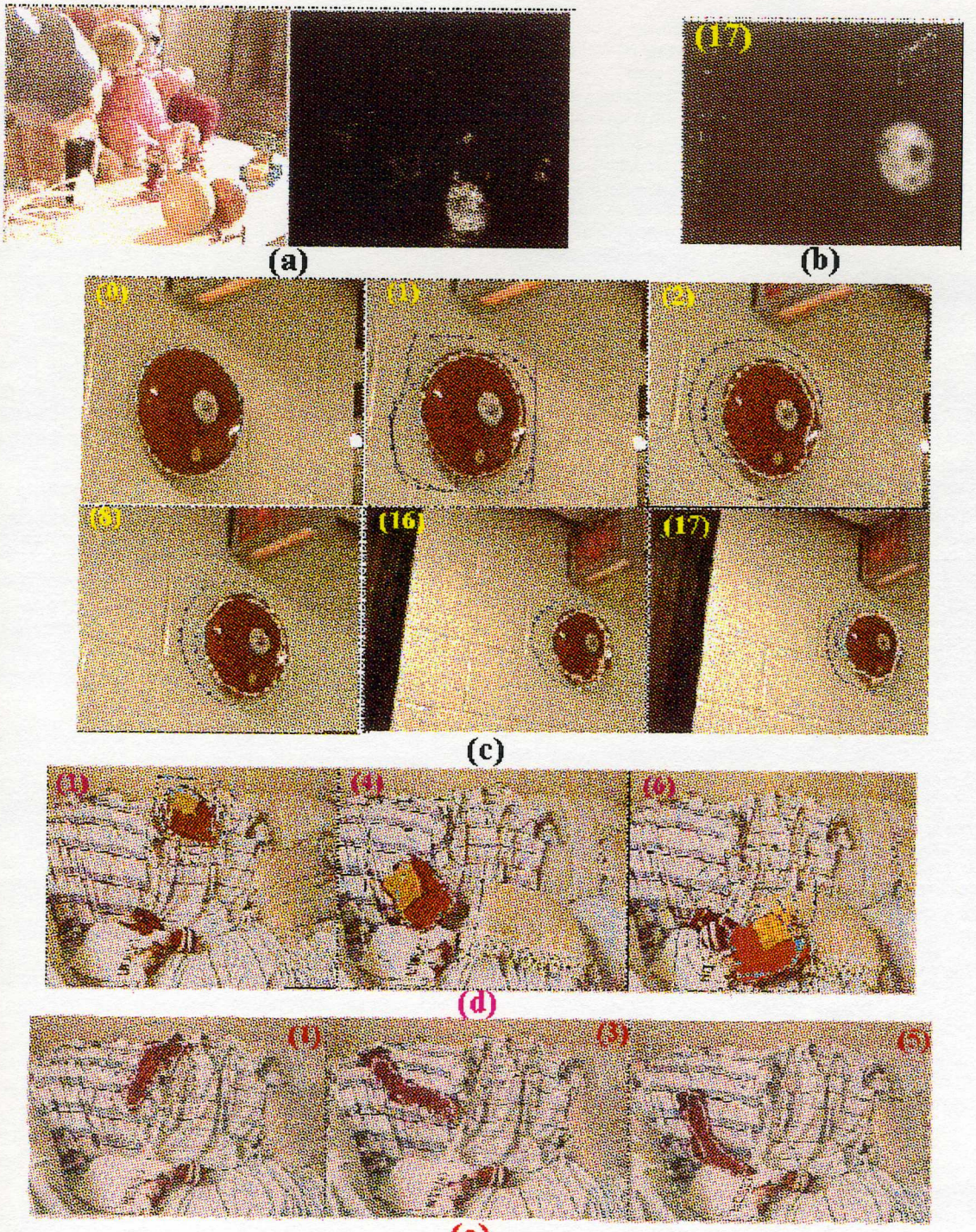

Image Plates: The closed contours in the images (c), (d) and (e) show the snake positions. The outmost contours are the initial snake positions and the inner most ones are the equilibrium positions. The countours in between are positions of the snakes during snake fitting. (a) A color image and its backprojected image using the pear as the model. (b) The backprojected image of image (17) of (c) using the bell in image (16) of (c) as the model.

One question is how the resolution of color space affects the performance of our tracking system. We measure the effect by looking at how good the backprojected images are under different resolutions of the color space. The less noise in the backprojected image near the instance of the target and the denser the blob(s), the better the backprojected image.

Our experiments show that if the image itself has poor color resolution (quality), then dividing the color space into finer buckets does not improve the backprojected image and it still tends to be noisy, unless the colors of the target are significantly different from those in background. On the other hand, if the image has good resolution, the finer the color space, the less noisy the backprojected image, but the weaker the response from the model in the backprojected image, too, unless the model has uniform surface color.

In our experiments, we only use the H and S axes in the IHS space, and we divide each of them into 100 units.

6.2 Target with Small Changes in Both Location and Size

Image Plate (c) shows images taken from a sequence of eighteen images on which we applied our tracking algorithms. The alarm bell in image (0) is selected as the model. The scale_ratio is 1.3 and d_size (defined in 4) is 0.3. These parameters indicate that the target moves slowly and its size changes gradually, too.

Also note even though the backprojected images are quite noisy (see Image Palte (b)), the tracking system still works well in this case.

6.3 Target with Regular Shape

Image Plate (d) shows three frames of images taken from a sequence of images. The red-yellow box in image (1) is selected as the target. The scale_ratio is 2.5. We observe that the snakes align reasonably well with the contour of the boxes.

This example shows that COTS works for objects with significantly different colors.

6.4 Target with Irregular Shape

Three frames of images taken from a sequence of images are shown in Image Plate (e). The red sock in the first image is chosen as the target. The scale_ratio is 3.0. Even though the contour of the socks in the images is irregular and the shape also changes significantly in the images, the snakes still align well along the contour of the socks. Note also that quite strong noise is present in the area of interest.

7 Conclusion

The major contribution of this paper is that it provides a framework for integration of surface color and active contour models to track color object. Color histogram backprojection has been used to extract the occurrences of the target in a sequence of color video images. This is accomplished by producing a backprojected image with high intensity values where colors of the target appear and low or zero intensity values everywhere else. Our experiments show that in practice a bright blob appears at the location corresponding to an instance of the target in the backprojected image while the rest of that image has very low intensity.

Our experiments also demonstrate that *snakes* are useful of finding the contour of the target in the backprojected image. The final position of the snake completely determines the shape and location of the blob, hence the location of the object being tracked is obtained. Color histogram matching is then used as a cue to verify the object. In case the snakes are prevented from settling on the target boundary by isolated high intensity pixels in the backprojected image, a method of region-splitting is used, which divides the region into two subregions, one of which contains the object and the other the isolated high intensity pixels. Then snake fitting is applied to subregion which has higher histogram matching value with the target.

A working interactive X11-based system, COTS, has been implemented on machines that allows users to display streams of images, define a model on an image, draw snakes, and execute other image and snake operations through a menu-based interface.

References

[1] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. In *1st IEEE Int. Conference. on Computer Vision*, London, 1987.

[2] Robert Massen, Thomas Regle, and Pia Böttcher. Color and shape classification with competing paradigms: Neural networks versus trainable table classifiers. *SPIE*, 1265 Industrial Inspection II, 1990.

[3] E. Milios, M. Jenkin, and J. Tsotsos. Design and performance of trish, a binocular robot head with torsional eye movements. *special issue on "Mobile robots, robot heads and active vision" of the International Journal of Pattern Recognition and Artificial Intelligence*, to appear, 1993.

[4] S. B. Nickerson, D. Long, M. Jenkin, E. Milios, B. Down, P. Jasiobedzki, A. Jepson, D. Terzopoulos, J. Tsotsos, D. Wilkes, N. Bains, and K. Tran. Ark: Autonomous navigation of a mobile robot in a known environment. In *International Conference on Intelligent Autonomous Systems, IAS-3*, Carnegie-Mellon University, Feb 1993.

[5] W. Pratt. *Digital Image Processing.* J. Wiley and Sons, 2nd edition, 1991.

[6] F. P. Preparata and M. Ian Shamos. *Computational Geometry: An introduction*, pages 106–110. Springer-Verlag, 2nd edition, 1985.

[7] Robert Sedgewick. *Algorithms*, pages 365–369. Addison-Wesley, 2 edition, 1988.

[8] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, Vol. 7, Nov. 1991.

[9] Ziqiang Wu and Evangelos E. Milios. On tracking color objects. Technical Report ARK project, in preparation, York University, Ontario, Canada, 1992.