Target tracking method based on a comparison between an image and a model *

Rafael Murrieta-Cid

Abstract

This report describes a method to track moving non-rigid objects. The method is based on a comparison between a model and an image, the comparison function employed is the partial Hausdorff distance. The model and the image are binary elements extracted from a sequence of gray levels images. The target's motion in a two-dimensional representation (image) is decomposed into two parts: a two-dimensional motion, corresponding to the change in the target's position in the image space and a two-dimensional shape change, corresponding to a new aspect of the target.

1 Introduction

The target tracking problem has received a good deal of attention in the computer vision community over the last years. Several techniques have been reported in the literature, and a variety of features have been proposed to perform the tracking.

Some methods attempt to track 3D objects by using multiple views [8]. Others use contours as features to perform the tracking [2], [7], [14]. Some contour methods use curve characterization [13] and active contours [6] to handle object deformations. Other approaches use either color information

*This research was funded by CONACyT, SEP, México and SFERE, LAAS France.

[16] to facilitate the tracking, or a combination of regions computed by color segmentation and contours [15]. Finally, some techniques identify motions by detecting variations of some global characteristic of the image such as texture or statistical invariants [11].

This report describes a method based on a comparison between a model and image. The model and the image are binary elements extracted from a sequence of gray levels images using a edge detector similar to [4]. The input data is handled as a configuration of binary elements, where the characteristic pixels (nonzero points) belonging to edges in the model and image are not required to be concatenated. The partial Hausdorff distance is used in order to measure the resemblance of the image with the model [10].

2 The target tracking method

This report describes a method to track moving non-rigid objects. The method is based on the assumption that the motion of a non-rigid object in three dimensional space can be characterized using a two-dimensional representation.

The target's motion in a two-dimensional representation (image) can be decomposed into two parts:

- A two-dimensional motion in the image, corresponding to the change in the target's position in the image space.
- A two-dimensional shape change, corresponding to a new aspect of the target.

The tracking is done using a comparison between an image and a model. It is possible to represent the model and the image with two binary arrays, where the (i, j) element in each array is 1 if it corresponds to an edge, or 0 otherwise. Figure 1 shows the original gray level image, and Figures 2 and 3 show the edges of both the image and model.

To reduce the number of operations, and consequently the time of computation, it is necessary only to search the model in a region contiguous to the target position in the previous frame of the sequence. This assumes a bound on the target's speed that guarantees that the target's new position

Figure 1: *original gray level image*

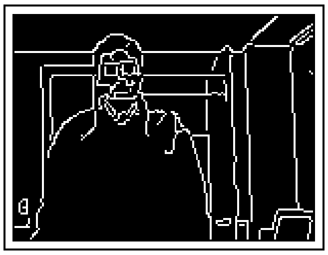

Figure 2: *edge's image*

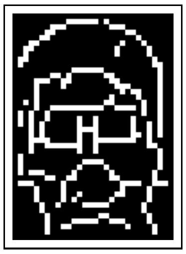

Figure 3: *model*

lies inside the region of examination. We also assume a known initial target position, and that the shape of an object does not change significantly between two successive frames of an image sequence.

3 Comparing the image with the model

To measure the resemblance of an image with the model we use the partial Hausdorff distance.

Given two sets of points P and Q, the Hausdorff distance is defined as (see [10])

$$H(P, Q) = \max(h(P, Q), h(Q, P))$$

where

$$h(P, Q) = \max_{p \in P} \min_{q \in Q} \parallel p - q \parallel \tag{1}$$

and $\parallel . \parallel$ is some norm for measuring the distance between two points p and q. The function $h(P, Q)$ (distance from set P to Q) is a measure of the degree in which each point in P is near to some point in Q. A small value of $h(P,Q)$ is implies that every point in P is close to a point in Q. The Hausdorff distance is the maximum among $h(P,Q)$ and $h(Q,P)$. Thus the Hausdorff distance measures the degree to which each point of P is near some point Q and vice versa.

By computing the Hausdorff distance in this way we are obtaining the most mismatched point between the two shapes compared; consequently, it is very sensitive to the presence of any outlying points. For that reason it is often appropriate to use a more general rank order measure, which replaces the maximization operation with a rank operation. This measure (partial distance) is defined as:

$$h_k = K^{th}_{p\in P} \min_{q\in Q} \parallel p - q \parallel \tag{2}$$

Where $K^{th}_{p\in P} f(p)$ denotes the K^{-th} ranked value of $f(p)$ over the set P. That is, if we consider the points in P to be in sequence ordered by their values $f(p_1) \leq \ldots \leq f(p_n)$, the K^{-th} element in this sequence, $f(p_K)$, is the K^{-th} ranked value. For example, the n_{-th} ranked value is the maximum (the largest element in the sequence), and the $n/2{-}th$ ranked value is the median.

One interesting property of the Hausdorff distance and the "partial distance" is the asymmetry inherent in the computation. The fact that every point of P (or subset of P) is near some point of Q says nothing about whether every point of Q (or subset of Q) is near some point of P. In other words, $h_{k1}(P,Q)$ and $h_{k2}(Q,P)$ can attain very different values. In fact each one of the two values give different information.

The term $h_{k1}(P,Q)$ is the unidirectional partial distance from the model to the image, and $h_{k2}(Q,P)$ the unidirectional partial distance from the image to the model. Where $P = M_t$ is the model and $Q = I_t$ is the image or region of the image given in at t time of one sequence. The maximum of these two values defines the partial Hausdorff distance.

4 Finding the model

The first task to accomplish is to define the position of the model M_t in the next image I_{t+1} of the sequence. The search of the model in the image (or image's region) is done in some direction selected.

The minimum value of $h_{k1}(M_t, I_{t+1})$ identifies the best "position" of M_t in I_{t+1}, under the action of some group of translations G. It is possible also to identify the set of translations of M_t such that $h_{k1}(M_t, I_{t+1})$ is no larger than some value τ, in this case there may be multiple translations that have essentially the same quality.

However, rather than computing the single translation giving the minimum distance or the set of translations, such that its correspond h_{k1} is no larger than τ, it is possible to find the first translation, such that its h_{k1} associated is no larger than τ, for one search direction given.

Although the first translation which $h_{k1}(M_t, I_{t+1})$ associated is less than τ it is no necessarily the best one, whether τ is small, the translation should be quite good; besides in the updating of the model (in order to renew the shape's change of the model), it is possible also to correct its position reducing the induced error. This is better than to compute all the set of valuable translation, whereas the computing time is significantly minor.

In order to compute the term $\min_{q \in Q} \parallel p - q \parallel$ of the equation (2) it is necessary to compute the distance from any position i, j in the image Q to the nearest point nonzero in this image. As the model is superimposed on the image, it computes the minimal distance between the model points (nonzero pixels) to the points in the image (also nonzero pixels). This distance has been referred as a distance transform, because it gives the distance from any point p to the nearest point in a set of source points Q, and it has also been called the Voronoi distance of Q by analogy to Voronoi diagrams, which specify the locations equidistant from two or more points of a given set.

4.1 The Voronoi distance $D_q(i, j)$

There are many methods of computing the Voronoi distance or an approximation of it [3], [5].

We use an algorithm similar to [12], which produces distance transform values that are exact up to machine precision for some norm of the distance.

This algorithm first processes each row independently. For each row of $Q(i, j)$, it calculates the distance to the nearest pixel in the row, for each (i, j) it finds Δi such that $Q(i + \Delta i, j) \neq 0$ or $Q(i - \Delta i, j) \neq 0$ and that Δi is the minimum nonnegative value for which this is true. It then computes the distances to the nearest pixel in each column in the same mode.

To reduce the number of operations the distance to the nearest point for a given position (i, j) is computed by comparing the distances of the possibles point candidates, just in the zone determined for the minimum value between the distance to the nearest pixel in the row, and the distance to the nearest pixel in the column. A look-up table that depends on the distance-norm is used to compute the value of the distances.

4.2 Two techniques that decrease the computation running time of the partial Hausdorff distance

The computation of the partial Hausdorff distance as a function of translation can take a significant time to run, because it considers every possible translation of the model until it is finds one where $h_k < \tau$.

The first technique consists of computing the number of points m of the model, which are at distance $h_k > \tau$ for a given translation; n is the total number of points in the model. If $m > n - k_{-th}$, it is not necessary to continue scanning the model, in view of the fact that it is not possible to have k_{-th} to a distance $h_k < \tau$; consequently, we can finish the analysis for that translation.

One second technique relies on the order in which the space of possible translations are scanned. Assume that the order is a row at a time in the increasing i direction, to superimpose the model on the image. Let $D'_{q(+i)}(i,j)$ be the distance in the increasing direction i to the nearest location where the Voronoi distance is $D_q(i,j) < \tau$. We can use $D'_{q(+i)}(i,j)$ to determinate how far we would have to move in the increasing i direction to find a place where $D_q(i,j) < \tau$. The model will translate at this location when $m > n - k_{-th}$ (see [9] for more details).

5 Updating the model

Having found the first translation g such that $h_{k1} < \tau$, we now have to build the new model M_{t+1} by determining which pixels of the image I_{t+1} are part of this new model. Figures 4, 5, 6 show different models after updating its shape.

The model is updated by using the distance $h_{k2}(I_{t+1}, g(M_t))$ as a criterion for selecting the subset of images points I_{t+1} that belong to M_{t+1}. The new model is defined by:

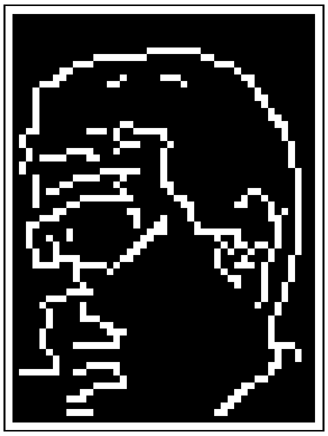

Figure 4: *model a*

Figure 5: *model b*

Figure 6: *model c*

$$M_{t+1} = \{q \in I_{t=1} \mid h_{k2}(I_{t+1}, g(M_t)) < \delta\} \qquad (3)$$

Where $g(M_t)$ is the model in the time t under the action of the translation g, and δ controls the degree to which the method is able to track objects that change shape. In the practice this is done by dilating $g(M_t)$ a δ in each direction (columns and rows). $\delta = 0$ will cause the tracker to lose an object after several frames, even if the object does not actually change due to noise; contrarily one too large δ could 'tracking' the background.

In order to allow for models that may be changing in size, this size is increased whenever there are a significant number of nonzero pixels near the boundary and is decreased in the contrary case. The model's position is improved according with the position where the model's boundary was defined.

The initial model must be computed with a different method because there is no previous model. The user specifies a rectangle in the frame that contains the model (see figure 7), and one image of the background is taken without the model. All the points belong to the background image and which

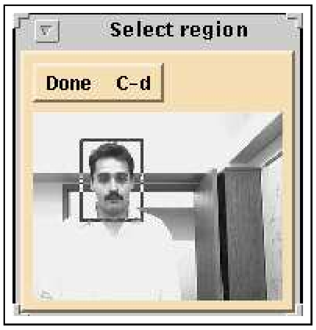

Figure 7: *Initial model*

aren't in the model are eliminate, in order to obtain a initial model without noise. It assumes that the camera does not move between the two pictures.

With this initial model the tracking is begun, finding progressively the new position of the target and updating the model. The tracking of the model is successful if:

$$k1 > fM \mid h_{k1}(M_t, I_{t+1}) < \tau \tag{4}$$

and

$$k2 > fI \mid h_{k2}(I_{t+1}, g(M_t)) < \delta \ , \tag{5}$$

in which fM is a fraction of the number total of points of the model M_t and fI is a fraction of image's point of I_{t+1} superimposed to $g(M_t)$.

6 Extensions over the general method

The target tracking method presented in this report is based in the one introduced in [9] and [10]. This section enumerates some of the extensions over the general method.

- Only a small region of the image is examined to obtain the new target position, as opposed to the entire image. In this way the computation time is decreased significantly. The idea behind a local exploration of the image is that if the execution of the code is quick enough, the new target position will then lie within a vicinity of the previous one. We are trading the capacity to find the target in the whole image in order to increase the speed of computation of the new position and shape of the model. In this way the robustness of the method is increased to handle target deformations, since it is less likely that the shape of the model changes significantly in a small δt. In addition, this technique allows the program to report the target's location to any external systems with a higher frequency (for an application see [1]).

- Instead of computing the set of translations of M_t, such that $h_{k1}(M_t, I_{t+1})$ is no larger than some value τ, we are finding simply the first translation whose $h_{k1}(M_t, I_{t+1})$ is less than τ. This strategy decreases the computational time. However, this technique could cause an accumulation of error that induces a tracking drift in a direction opposite to the search. It is possible to avoid drifting by resizing the target bounding box once its new position is computed, and by reducing the tolerance of the model and image mismatch.

- In order to compute the Voronoi distance $D_q(i, j)$, we use an algorithm which produces distance transform values that are exact up to machine precision. The algorithm structure is the same for any distance metric $D[(x_1, y_1), (x_2, y_2)]$ that is a function of the relative values of the pairs (x_1, x_2) and (y_1, y_2), is nondecreasing in $|x_1 - x_2|$, and nondecreasing in $|y_1 - y_2|$ (e.g., city block, chess-board and Euclidean distances).

7 Conclusion and future work

In this report we have described a target tracking method that basically consists of a comparison between an image and a model. The comparison function employed is the partial Hausdorff distance. Several techniques that decrease computation time were presented.

The basic ideas of our implementation are the following:

- The image of a solid object moving in space can be decomposed into two components: two-dimensional motion and a two-dimensional deformation.
- The search of the target is done only in a vicinity of the image near to the previous target location, instead of the entire space image.
- We find simply the first translation for which unidirectional Hausdorff distance $h_{k1}(M_t, I_{t+1})$ is satisfied.

This method was implemented in C and C++, and the code is capable of processing a frame in about 0.3 seconds. Processing includes smoothing, edge detection, target localization, and model updating for a video image of (160 X 120 pixels) on a SPARC 20.

Figures 9, 11 and 13 show the evolution of the model for a sequence of gray levels images.

Figure 8

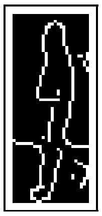

Figure 9

Figure 10

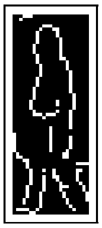

Figure 11

Figure 12

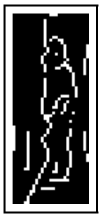

Figure 13

Figures 14, 15, 16 and 17 show different snapshots of a target tracking experiment executed in real time. The images correspond to the tracking of a person's head, which is marked in the figure with bounding box.

In terms of adding functionality to the system, there are several possible extensions:

First, we are extending the algorithm in order to handle different models (different views) of a single target. The objective is to improve robustness toward changes in shape, and to allow target recovery in case of loss.

Second, we plan to study image preprocessors that would enhance those image features that are appropriate to our method.

Third, currently the user specifies the location and size of the feature in the initial frame to specify the model to track. We would like to explore the possibility of an automatic feature identification system. This system should use several attributes of the image (such as color, texture, shape) in order to select the initial model.

Figure 14

Figure 15

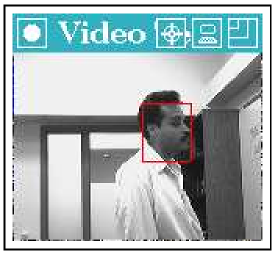

Figure 16

Figure 17

Finally, we would also like to consider the case of multiple targets.

References

[1] Becker C., González-Baños H., Latombe J.-C. and Tomasi C. "An intelligent observer.", In *Proc. International Symposium on Experimental Robotics*, 1995.

[2] Blake A. et Al "Affine-invariant contour tracking with automatic control of spatiotemporal scale", *4th International Conference on Computer Vision*, (USA: IEEE Computer Society Press, April 1993, p. 66-75).

[3] Borgefors G. "Distance transforms in digital images", IEEE *Transactions on pattern analysis and machine intelligence*, Vol. 8, 1986, p. 344-371.

[4] Canny J. F. "A Computational Approach to Edge Detection", IEEE *Transactions on pattern analysis and machine intelligence*, Vol. 8, n°6, 1986, p. 679-698.

[5] DANIELSSON P. E. "Euclidean distance mapping", *Comput. graphics Image processing*, Vol. 14, 1980, p. 227-248.

[6] DELAGNES P., BENOIS J. and BARBA D. "Adjustable polygons: a novel active contour model for objects tracking on complex background", *Journal on communications*, Vol 45, 1994, p. 83-5.

[7] GARIBOTTO G. "Motion tracking of connected edge contours", *Time-varying image processing and moving object recognition. proceedings of the 3rd international workshop*, 1990. p. 323-30.

[8] GAVRILA D.M. et Al "3-D model-based tracking of humans in action: a multi-view approach", In *Proceedings of* IEEE *Conference on Computer Vision and Pattern Recognition*, 1996, p. 73-80.

[9] HUTTENLOCHER D.P., NOH J.J. and RUCKLIDGE W.J. " Tracking non-rigid objects in complex scenes", In *Proceedings Fourth International Conference on Computer Vision*, 1993, p. 93-101.

[10] HUTTENLOCHER D.P. et Al. "Comparing images using the Hausdorff distance", IEEE *Transactions on pattern analysis and machine intelligence*, Vol. 15, n°9, Sept 1993, p. 850-863.

[11] JIANSHO S. and TOMASI C. "Good features to track", *Proceedings of* IEEE *Conference on Computer Vision and Pattern Recognition*, 1994, p. 593-600.

[12] PAGLIERONI D.W. "Distance transforms: properties and machine vision applications", CVGIP *Graphical models and Image processing*, Vol 54, n°1, Jan 1992, p. 56-74.

[13] REHG J. M. and KANADE T. "Visual tracking of high DOF articulated structures: an application to human hand tracking", In *Proceedings of Third European Conference on Computer Vision*, Vol II, 1994, p. 35-46.

[14] SAGE D. et Al "A contour feature tracking system", *Proceedings of the 7th international conference on robot vision and sensory controls:* ROVISEC-7 *advanced sensor technology*, UK: IFS (Publications), 1988. p. 247-56.

[15] Wren C. et Al "Pfinder: real-time tracking of the human body", In *Proceedings of the* SPIE *International society for optical engineering*, Vol 2615, 1996, p. 89-98.

[16] Yue Du et Al "A color projection for fast generic target tracking", In *Proceedings of* IEEE/RSJ *International Conference on Intelligent Robots and Systems*, Vol 1, 1995, p. 73-80.