General Purpose Real-Time Object Tracking Using Hausdorff Transforms

David Vignon[‡†], Brian C. Lovell[†], and Robert J. Andrews[†]
[†]Intelligent Real-Time Imaging and Sensing (IRIS) Group
The School of Information Technology and Electrical Engineering
The University of Queensland, Australia QLD 4072
[‡†] and École Polytechnique de l'Université de Nantes, France
vignon_david@yahoo.com, {lovell, andrews}@itee.uq.edu.au

Abstract

We describe a real-time computer-vision tracking module capable of using several Hausdorff distance based approaches to localize and match edge models in a scene. The implementation is based on widely supported software and hardware technologies such as Microsoft DirectX/DirectShow, Intel Image Processing and the Open Source Computer Vision libraries.

Keywords: Computer Vision, Hausdorff Distance, Object Tracking

1 Introduction

The system described in this paper is based on the object tracking methods developed by Huttenlocher *et al* [1]. This system can track multiple non-rigid objects, such as people, in a cluttered environment, such as an office or laboratory. In 1995 a similar real-time tracking system was implemented by Welsh and Ahmed [3] using a cluster of Sun workstations with shared- and distributed-memory multiprocessors connected via high-speed ATM links. The main challenge of this project was to implement a real-time system on a garden-variety standalone PC running Windows 2000 by using real-time image processing software technologies such as Microsoft DirectX/DirectShow, the Intel Image Processing library, and the Intel Open Source Computer Vision library.

2 The Hausdorff Distance Measures

Hausdorff distance is a scalar measure of the distance between two sets of points. In practice, the two sets of points may be obtained by edge detecting a reference image and a target image. The Hausdorff distance measure can the be used to determine the current position of the selected object within the image.

Consider the interpretation of the distance of a single point X from a set of points A. When we say that X is a distance d from A, d is often considered to be the Euclidean distance from X to the nearest point of A. The Hausdorff distance naturally extends this concept to the distance between two sets of points, A and B say. If we determine the distance of each point in B from a set of points A as above, we will then have N Euclidean distance measures, where N is the number of elements in B. As we want a *scalar* measure of distance, we choose the maximum value of these distance measures, known as the *directed* Hausdorff distance.

2.1 The directed Hausdorff distance

Formally, the *directed* Hausdorff distance is defined as

$$h(A,B) = \max_{a\in A} \min_{b\in B} \left\| a - b \right\|. \qquad (1)$$

where $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$ are two sets of points, and $\|.\|$ is the distance between points a and b measured by some norm (generally the Euclidean norm L_2). It identifies the point $a \in A$ that is farthest from

any point of B, and measures the distance from a to its nearest neighbour in B. Thus $h(B, A)$ is commonly referred to as the *forward Hausdorff distance* between A and B, while $h(A, B)$ is the *reverse distance*. One interpretation of the forward distance $h(B, A)$ is the distance *within* which we can find every point of A from B and similarly for the reverse distance.

The forward and reverse distances are not necessarily the same. For example, consider the sets of points which comprises the graphic symbols '+' and '-' and further assume that these symbols are aligned. The set of points corresponding to the symbol '-' is a subset of the set of points representing '+' so the distance is zero, but the reverse distance is definitely positive and is, in fact, equal to half the height of the '+' symbol. To create a symmetric distance measure we define the *total Hausdorff distance*, H, as the maximum of the forward and reverse distances. That is

$$H(A, B) = max\Big\{h(A, B), h(B, A)\Big\}. \qquad (2)$$

The lower the distance value is, the better the match.

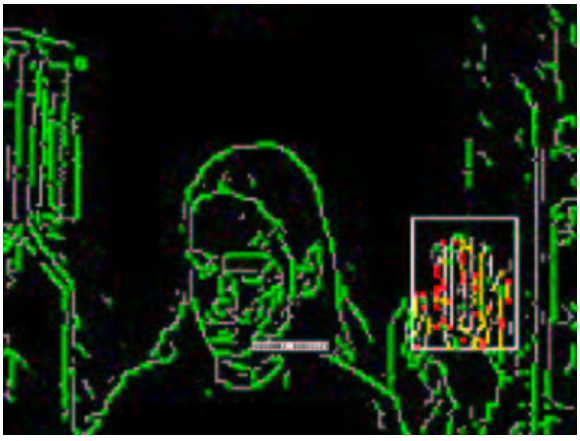

Figure 1: Example of the system being used for hand tracking

2.2 The generalized Hausdorff distance

Now the directed distance $h(A, B)$, and hence the Hausdorff distance, will be small when every point of A is near some point of B. Thus this distance is a very fragile measure because it may depend on just a single outlier pixel in the entire image causing $h(A, B)$ to be large. This is common particularly when the object is partially occluded, or noisy due to poor edge extraction or capture. The definition of equation 1 is thus replaced with a more robust generalization of the Hausdorff distance based on order statistics, sometimes called the "*partial Hausdorff distance*" by taking the k^{th} ranked distance rather than the maximum distance. Thus *generalized Hausdorff distance* is defined by

$$h_k(A, B) = k^{th}_{a \in A} \min_{b \in B} \Big\|a - b\Big\| \qquad (3)$$

where k^{th} denotes the k^{th} largest value.

2.3 Application to object tracking

Most object recognition systems require a similarity measure between the model (or reference) features and the image features. The Hausdorff distance measures the divergence of a set of features with respect to a reference set of features [1]. The features are usually image edges in practice. In general, we are interested in using the Hausdorff distance to identify instances of a model in an image or to track a moving object in a scene.

Now if A denotes the set of scene points and B denotes a set of model points, let t be some transformation of B in the 2D space; only translation and scaling are considered here — separately in x and y. The values of t where $h\Big(t(B), A\Big)$ is small are the most likely transformations of the model. Thus for a range of values of t, $h\Big(A, t(B)\Big)$ is evaluated to determine the minimum distance in the transform search space.

3 Real-Time Imaging Software Tools

Microsoft® DirectX® is a set of low-level application programming interfaces (APIs) for creating games and other multimedia applications. Microsoft DirectShow is an API for streaming media on the Microsoft Windows®

platform. DirectShow simplifies media playback, format conversion, and capture tasks. At the same time, it provides access to the underlying stream control architecture for applications that require custom solutions.

The Intel® Image Processing Library provides a set of low-level image manipulation functions in standard DLLs and static libraries form. The functions are optimized for Intel Architecture processors, and are particularly effective at taking advantage of MMX (Multimedia Extensions) technology, the Streaming SIMD Extensions (SSE) and SSE-2 [6]. The Open Source Computer Vision Library is mainly aimed at real-time computer vision [7].

4 Efficient Implementation

There are several techniques for computing the Hausdorff distance and we needed to determine the method best matched to our machine architecture.

4.1 Dilation method

Given that the forward distance $h(B, A)$ is the distance within which we find every point of A from B (or $t(B)$), the most both straightforward and intuitive method is to use successive dilations.

Let A' be the dilation of the scene A by a disc C_δ of radius δ, m the number of points of the model B, k the number of points of B which should be aligned with A' (partial Hausdorff-distance), p the number of points of $t(B)$ covered by A'. We refer to $\frac{p}{m}$ as the Hausdorff-fraction and $\frac{k}{m}$ as the threshold fraction. Intuitively, $\frac{p}{m}$ measures the percentage of $t(B)$ which lies within distance δ of the points of A.

Thus we search for the minimum radius δ by which the model set A must be dilated to cover $\frac{k}{m}$ of the points in B. This calculation could involve many dilations of the reference image A which is computationally quite expensive.

4.2 Distance transform method

In practice, it is extremely time-consuming to dilate the scene A (or a portion of it) many times for all transformations, and then to dilate $t(B)$ for those times where we also need to compute the reverse Hausdorff distance. It is far more efficient to compute once only the distance transform (using Intel IPL library) of the scene and to then threshold it to produce the various dilated model sets. Similarly, we compute once only the distance transform of the scaled model and we translate and threshold it. For all pixels of the image, the distance transform gives the distance to the closest edge-pixel. For example, If this transform is thresholded at 1 (if value ≤ 1 then 1 else 0), this is equivalent to dilating with a disk of radius 1.

Note that finding the crossproduct between the binary images is just simple logical AND between A' and $t(B)$ which determines p.

4.3 Hausdorff Fraction tracking

In a real-time system, it is sensible to fix the Hausdorff distance to a particular value δ, and use this value as the radius of dilation for the point sets A(and also for $t(B)$ for the other direction of the Hausdorff measure).

The *forward* and *reverse Hausdorff fractions* between the two sets are then calculated as [3]:

$$\begin{aligned} h_k\Big(t(B), A\Big) &= \%\text{ of t(B) which align } A + C_\delta \\ &= \frac{p}{m} \end{aligned}$$

$$\begin{aligned} h_k\Big(A, t(B)\Big) &= \%\text{ of A which align } t(B) + C_\delta \\ &= \frac{q}{n} \end{aligned}$$

where C_δ is a disc of radius δ.

Let be $t(B)'$, the dilation of the model $t(B)$, n the number of points of the scene in the location of $t(B)$ (Region of Interest) and q the number of points of A correlated with $t(B)'$. Thus $\frac{k}{n}$ is the threshold fraction for the reverse Hausdorff fraction.

Practically, this can be implemented by performing the distance transform, thresholding at δ, performing a logical AND followed by counting pixels.

Note: here, the total Hausdorff fraction is given by the minimum of the reverse and forward fraction (while the total Hausdorff distance is given by the maximum of the forward and reverse distance). The best transformation is the transformation for which the Hausdorff fraction is largest (rather than smallest for a Hausdorff distance-based method).

4.4 Assumptions on the inter-frame movement of the model

In the case of object tracking for a video sequence, we assume that motion and shape change from frame to frame are small, thus some optimizations limit the space of search to a small range of translations and scalings from the previous model position.

4.5 Multiresolution techniques

Huttenlocher describes a multi-resolution method for scanning the space of possible transformation of a model based on the partial Hausdorff distance. This approach makes the reasonable assumption that if $H\Big(A, t(B)\Big)$ is large for certain values of t, it must also be large all nearby values t' [2]. This allows large portions of the transformation space to be discarded efficiently by quantising the space it into searches at different resolutions.

For the forward distance, the distance transform of the image set A is first computed, and then probed at the locations of transformed model points. The values at those coordinates are sorted and $h\Big(t(B), A\Big)$ is given by k^{th} the value.

Practically, ranking is very time-expensive, so we exploit thoroughly the tools available in the image-processing dedicated libraries:

- The distance transform of A is performed and masked,
- Then we compute the histogram of the masked image and use the cumulated histogram method to sort out the resulting pixel. The Hausdorff distance is the index at $\frac{k}{m}\%$ of the maximum value of the cumulated histogram,
- We use an image pyramid of the scene and the model as a multi-resolution method. Actually, a transformation (*eg.* translation) at a coarser resolution of the pyramid is equivalent to several transformations at a finer resolution,
- Finally, we keep the best match instead of the best matches for testing in a finer resolution of the pyramid, contrary to Huttenlocher.

In the same way, we compute the distance transform of $t(B)$ and then scan this array at the locations of the points of A.

4.6 Relation to Chamfer methods

The Chamfer system is a related shape-based object detection [4] technique which has been used to great effect by Garvrila *et al* [5] in detecting road signs and pedestrians from moving vehicles. Basically, the Chamfer matching measure is the crossproduct of the distance transform of the edge-detected scene and the transformed model. The lower the value is, the better the match. If, for example, the average distance value lies below a certain threshold, the target object is considered detected.

When applied to object tracking, the Chamfer System can be seen as a simplification of the previous method, avoiding ranking (even if we need to examine both forward and reverse transform for the tracking application rather than the object recognition application). Thus the same method as above is applied (distance transform, followed by logical AND) but an average of resulting pixels is performed instead of a cumulated histogram.

4.7 Model Refreshing

Even if we assume that the model undergoes small transformations from frame to frame,

the cumulative changes can be too great to match the object in the current frame with the original model. We need then to update the model once the Hausdorff distance grows too large (or the Hausdorff fraction becomes too small). This adaptive process allows us to track non-rigid objects which undergo significant deformations through the scene. An example tracking despite such deformations is shown in Figure 2.

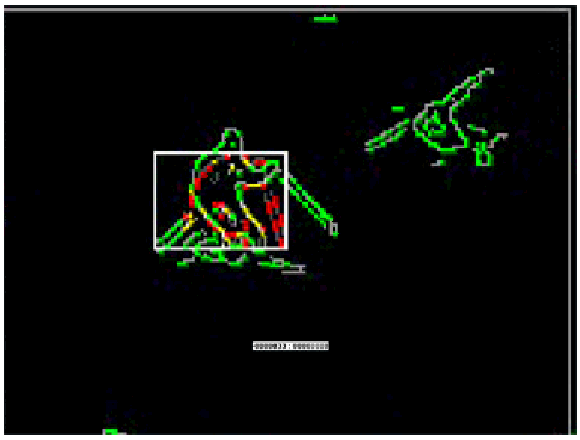

Figure 2: Output of the system tracking a highly deforming object (a skier)

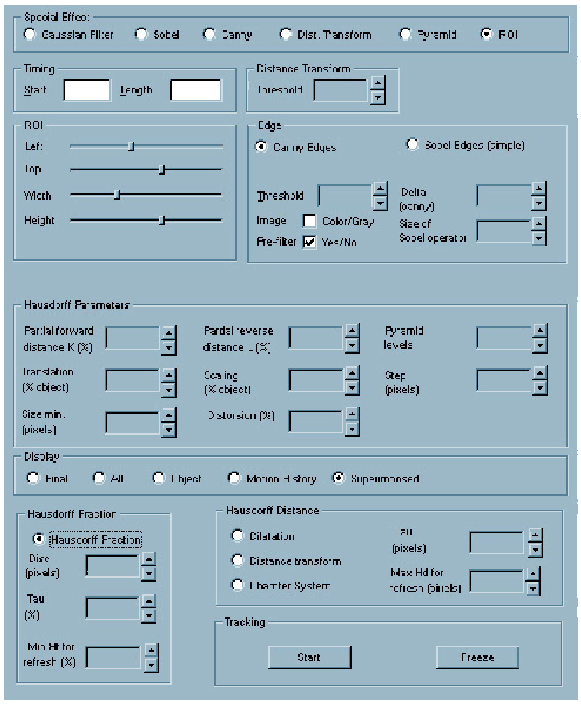

Figure 3: View of the tracking system property page

5 The Final System

The final system was made available as an AX file which could be loaded into the filter graph editor application of the DirectX SDK. A property page dialog (as shown in Figure 3) is used to control the operation of the tracker module. The final system was able to track objects such as hands and faces at about 12 frames per second. While the current system only tracks one object at time, there is no real difficulty in extending the method to track multiple objects apart from the computational overhead. Video outputs of the working system are available from `http://www.itee.uq.edu.au/~iris/ComputerVision/videos.html`.

6 Conclusions

It is apparent that with careful coding and appropriate software technologies, it is now possible to produce complete computer vision tracking systems that run on a standalone PC. Such tracking systems will form part of the core technology for future applications in video annotation and passive surveillance within the research group.

References

[1] D.P. Huttenlocher. *Comparing Images Using the Hausdorff Distance*. IEEE Transactions on PAMI, 15(9), September 1993.

[2] Daniel Huttenlocher & William Rucklidge. *Hausdorff-Based Image Comparison - A multi-Resolution Technique for Comparing Images Using the Hausdorff Distance* -. Cornell University. December 1992. `www.cs.cornell.edu/vision/hausdorff/hausmatch.html`

[3] Matt Welsh and Nawaaz Ahmed. *Real-Time Hausdorff-Fraction Motion Tracking on a Clustered Workstation*. 25 November 1995. Computer Science Department, Cornell University. `www.cs.berkeley.edu/ mdw/projects/rt-hausdorff/rt-hausdorff.html`

[4] Dariu M. Gavrila. *The Chamfer System.* `www.gavrila.net/Computer_Vision/Chamfer_System/chamfer_system.html`

[5] Dariu. M. Gavrila and V. Philomin, *Real-time Object Detection for Smart Vehicles.* Proc. of IEEE International Conference on Computer Vision, Kerkyra, Greece, 1999.

[6] *Image Processing Library Reference Manual.* `developer.intel.com/software/products/perflib/ipl/index.htm`

[7] *Intel Open Source Computer Vision Library Reference Manual.* `www.intel.com/research/mrl/research/opencv/`