An Object Recognition Scheme Using Knowledge and the Hausdorff Distance

C. Tzomakas and W. von Seelen

Institut für Neuroinformatik
Ruhr Universität Bochum
44780 Bochum, Germany

Abstract

In this paper we show how we can use a priori *knowledge from our environment in order to build a robust and efficient recognition scheme. The knowledge we use comes from physical constraints and the camera geometry. The recognition process is based on the* Hausdorff distance *transform. We propose a modification to the algorithm described in [6] that accomplishes the elimination of the scaling variance.*

Keywords: *distance transforms, perspective geometry, object recognition*

1 Introduction

The geometric comparison of contours is a very favorable approach in the object recognition community. Most of the model-based schemes in object recognition use distance measurements between the model features and the image features in order to extract a correspondence between them ([1], [11]).

The *Hausdorff distance* is a distance transform that has been widely used for such comparisons with successful results ([5], [6]). It provides a measure of the similarity between two sets of points and copes robustly with noise, too. In [7] the Hausdorff distance is further applied to the tracking of non-rigid objects in image sequences. In the rule two-dimensional models are extracted from the image data and matched to succesive frames of the image sequence according to this distance measure. In general the search space consists of all the possible affine transformations (translation, scaling and rotation) that the model has to undergo in order to fit into the scene.

However, in these aproaches the camera is static. If *ego-motion* exists and without knowledge about the form and the expected size of the model, the separation of a moving object from its background will be quite difficult. Furthermore, the algorithm proposed by Huttenlocher ([6]) works efficiently in the case where the size of the object is expected to vary over a known small range of scales. Otherwise, the search space explodes and the search becomes inefficient.

In our case now, we apply the Hausdorff transform to the analysis of traffic scenes. The scenes were acquired from a camera installed in a vehicle while driving on a highway. The goal is to detect foregoing vehicles and identify them. So the objects we want to recognize are rigid (e.g. cars). The rotation factor in our samples is very small, so that it can be ignored. We actually have to cope with all possible translations and scalings that can bring our model near to an object in the scene so that it fits it. But still the dimensions of the search space can be prohibiting for the aim of a real time application. We need some cues that will reduce the complexity of our task.

In the next section we give the mathematical definition of the Hausdorff distance and briefly present the *distance transform* which is used for the computation of the Hausdorff distance. A detail description of the distance and its properties can be found in [5]. In section 3 we describe our system's geometry and explain which kind of knowledge we use in order to shrink our search space. We also provide a solution for the estimation of the size of an object in the image. The search algorithm that calculates the Hausdorff distance over one frame is presented in section 4, and we conclude with some results and the discussion.

2 The Hausdorff distance

Given two finite point sets $P = \{p_1, \ldots, p_m\}$ and $Q = \{q_1, \ldots, q_n\}$, the Hausdorff distance is defined as

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

where

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

and $\|\cdot\|$ can be any norm (usually the *Euclidean* norm is used).

The functions $h(P,Q)$ and $h(Q,P)$ are called the *directed* Hausdorff distances and in general they are not symmetric. Assuming that P is the model-point set and Q the image one, we call them the *forward directed* and the *backward or reverse directed* Hausdorff distance respectively.

Thus, $h(P,Q) = d$ means that each point of set P must be within distance d of some point of Q, and also at least one point of P is exactly at distance d from the nearest point of set Q.

Another useful definition is the *partial* directed distance which allows the comparison of *portions* of two sets. This will allow the recognition of objects that are either partly occluded or distorted due to noise in the input data.

The idea is to rank each point of set P according to its distance to the nearest point in set Q, and take the Kth ranked element instead of the maximum. Thus, the definition of distance for K of the m model points $(1 \leq K \leq m)$ is:

$$h_K(P,Q) = K^{th}_{p \in P} \min_{q \in Q} \|p - q\|.$$

2.1 The distance transform

From the definition of the Hausdorff distance we have:

$$H(P,Q) = max\left(\max_{p \in P} \min_{q \in Q} \|p - q\|, \max_{q \in Q} \min_{p \in P} \|q - p\|\right).$$

If we define $d(x) = \min_{q \in Q} \|x - q\|$ *and* $d'(x) = \min_{p \in P} \|p - x\|$, we have

$$H(P,Q) = \max\left(\max_{p \in P} d(p), \max_{q \in Q} d'(q)\right)$$

that is, the Hausdorff distance $H(P,Q)$ can be obtained by computing $d(p)$ and $d'(q)$ for all $p \in P$ and $q \in Q$, respectively. The graph of $d(x)$, $\{(x, d(x)) | x \in \Re^2\}$ is a surface that is called the *Voronoi surface* of a set P ([8]). It has also been referred to as a *distance transform* (e.g. [3]) because it gives the distance from any point x to the nearest point in a set of source points P. Fig. 1 illustrates an example image of contour points (boxes) and a topdown view of the corresponding Voronoi surface, where brighter portions of the surface correspond to bigger distances to the contour points.

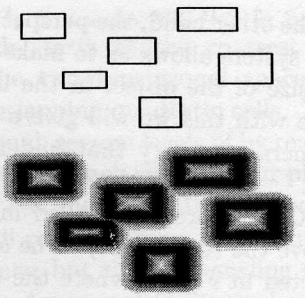

Figure 1: Set of contour points and a corresponding Voronoi surface

In other words, once the distance function of a point set Q is known, the directed Hausdorff distance of another point set P to Q is just a maximum operation over the positions indicated by the points of set P. The computation of the distance function of a set Q can be performed in $O(q)$ due to specialized graphics hardware for rendering and z buffering ([8], also [1], [2]).

3 Camera geometry and *a priori* Knowledge

As already mentioned, we may use some information from our environment to increase the performance of the recognition task.

First of all we have knowledge about the form and the physical size of the objects we want to match. The widths of cars for example vary only in a small amount in the real world (e.g. typical widths of cars range in $w = [1.6m - 1.8m]$). Furthermore there is a relation between the height and the width of a car (regularly the height of a car is a few less than its width). These remarks have led us to the result of using one general model to match all the objects of the category "cars". A typical model in our approach looks like the one in Fig. 2.

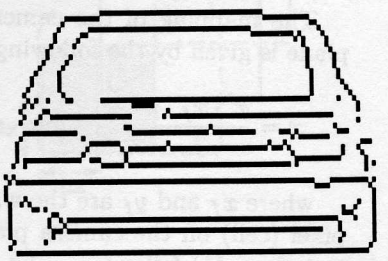

Figure 2: Example model of a car extracted from the segmented image

On the other hand, the perspective geometry of our camera system allows us to make a prediction for the actual size of the object in the image plane. Before we cope with this we will give a short description of the camera geometry that underlies our recognition scheme.

A CCD camera is installed in a vehicle at height H (above the road plane). The stance of the camera is depicted in Fig. 3, where the tilt is taken zero for reasons of simplicity and the camera plane is aligned to the (x, z) plane along to the z axis. These assumptions can be done without loss of generality.

In Fig. 3, (x, y, z) denote the world coordinates, (x', y') are the image coordinates, and f is the focal length of the camera.

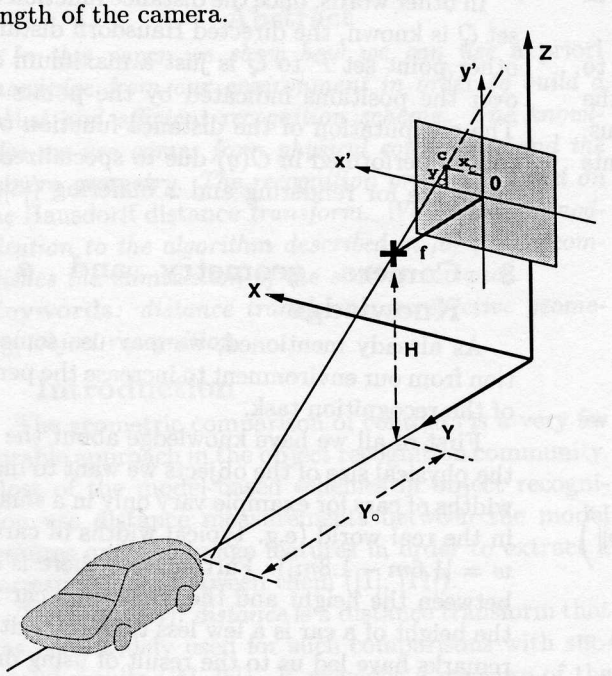

Figure 3: The camera geometry

The mapping of the camera plane to the ground plane is given by the following equations:

$$x = \frac{x' \cdot x_f \cdot y}{f} \qquad \textit{and} \qquad y = \frac{f \cdot H}{y' \cdot y_f} \tag{1}$$

where x_f and y_f are the width and the height of a pixel (cell) on the camera plane. Usually: $x_f = y_f$, thus from (1) follows:

$$x' = \frac{x.y'}{H} \tag{2}$$

Now for two points (left and right edge of the rear view of a car) we have:

$$x'_1 - x'_2 = \frac{(x_1 - x_2) \cdot y'}{H} \Rightarrow w' = \frac{w \cdot y'}{H} \tag{3}$$

where w denotes the width of the object in the real world and w' its width in the camera plane.

This means that the width of the object's idol in the image plane is proportional to the vertical distance of its lower edge to the optical axis. This is straight analogous to the fact that the size of an object perceived by the camera is related to its distance to the camera.

Thus, since the height of a car is proportionally related to its width, the vertical position of the idol in the image plane (i.e. the camera plane) gives an estimation of its size (see Fig. 4).

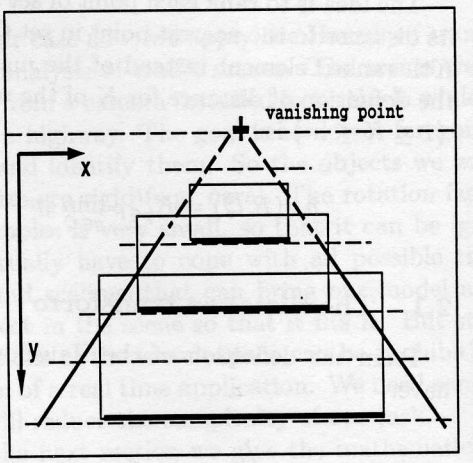

Figure 4: The size of the object's idol depends on its vertical position in the image

The main assumption in our recognition scheme is that the translatory surface is planar. When this condition holds, the optical axis equals to the vanishing point of the image. The vanishing point is calculated by a statistical contour based algorithm described in [10]. Similar approaches have been used in [12], [13].

4 The Algorithm

The algorithm is based on the multi-resolution approach described in [6]. The goal is to search efficiently in the transformation space for a translation and a scaling value for the model that will bring it arbitrary close (e.g. below a threshold distance τ) to a part of the image, and thus perform a matching.

In [6] the transformation space is a set of translation and scale values (t_x, t_y, s_x, s_y), thus four-dimensnional, and covers all the possible instances of

the model in the image. This space is iteratively quantized into equally-sized non-overlapping cells at different levels of increasingly fine resolution.

Starting from a top level of very coarse resolution and proceeding downwards, the forward Hausdorff distance of a fraction f of the model contour points (i.e. the *partial* distance) with respect to the image contour points is computed at the center of each cell for all the cells of the first level. The transformation cells of one level which satisfy a certain criterion give an hypothesis for the existence of the model and therefore are the candidates for the next resolution level. At the last level with the finest resolution the remaining cells are verified with the reverse distance (i.e. the distance from an image part to the model).

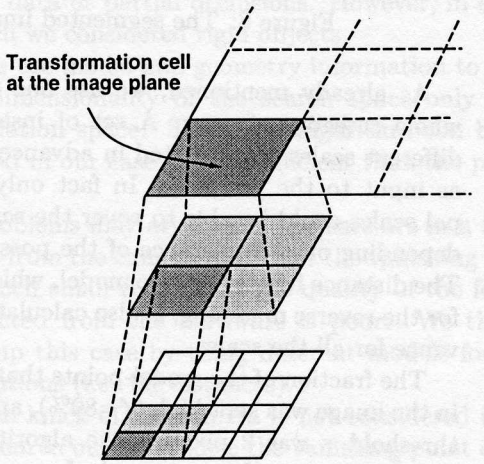

Figure 5: Quantization of the transformation cells

The main difference from Huttenlocher's approach in our case is that, as we see from eq. (3), the size of an object depends on its vertical position in the image. Furthermore, we are dealing with rigid objects and knowing their shape we have a dependence between the s_x and s_y scalings. Thus, in practice we reduce the four-dimensional search space to a two-dimensional one (i.e. the translation space), since at each translation pair (t_x, t_y) the scaling of the model will be known. In other words, we perform the matching by translating the model over the image and simultaneously scaling it at the right size.

In more detail now, the translation space is scanned from left to right, and from bottom to top in a multi-resolution way, as in [6]. Nevertheless, in our case the translation space initially does not have to cover the whole image, we rather search in the vertical direction from the lowest row of the image up to the height of the vanishing point (that is: $0 \leq t_y \leq y_{vanish}$ and $0 \leq t_x \leq x_{max}$).

The search starts at the coarsest resolution by quantizing the two-dimensional space into equally-sized non-overlapping quadratic cells. We proceed iteratively through several levels of increasingly fine resolution by dividing each cell into four pieces (quadtree partitioning, see Fig. 5). In fact, each cell of our process is still representing a four-dimensional transformation space, but since the scaling parameters are predicted it is drawn as two-dimensional.

At each level of the refining process we compute the partial Hausdorff distance of the model (translated to the middle of the cell and with the appropriate scale) and compare it with a threshold which depends only on the size of the cell, but not its location. In fact, considering a cell at level i representing a rectilinear region in the transformation space: $R = [t^i_{x_A}, t^i_{x_B}] \times [s^i_{x_A}, s^i_{x_B}] \times [t^i_{y_A}, t^i_{y_B}] \times [s^i_{y_A}, s^i_{y_B}]$, and letting d_{x_i} and d_{y_i} denote the dimensions of this cell with:

$$d_{x_i} = \left(\lceil (t^i_{x_B} - t^i_{x_A})/2 \rceil + \lceil (s^i_{x_B} - s^i_{x_A})/2 \rceil\right)$$

and

$$d_{y_i} = \left(\lceil (t^i_{y_B} - t^i_{y_A})/2 \rceil + \lceil (s^i_{y_B} - s^i_{y_A})/2 \rceil\right),$$

then the partial distance computed at the center of the cell is checked against the threshlold: $\tau'_i = \rho_i + \tau$, where ρ_i denotes the distance from the center of the cell to its corners ($\rho_i = \|d_{x_i}, d_{y_i}\|$).

If it is bigger than this threshold, then the whole set of translations included in this cell is of no interest for the further search. Otherwise the cell is considered as a possible region that could contain an instance of the model (see Fig. 6).

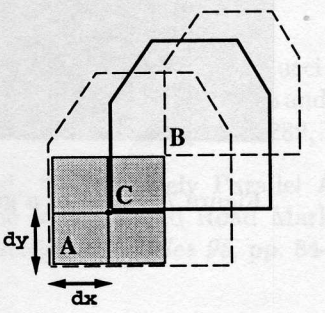

Figure 6: Model translated to the center of the cell

We proceed to the next level of higher resolution searching only the cells that at the current level did not exceed the threshold for that level. At the finest

resolution where each cell contains only one translation pair we compare the partial Hausdorff distance with the threshold τ. We also verify the results of the latest level with the reverse distance, in order to exclude some false matches that may arise due to noise.

At the beginning all the cells are marked as interesting. For the efficiency of the computations we also apply the techniques described in [6] (early rejection, early acceptance, skipping forward) that speed up our search process without missing any match. In the same citation the reader can find the reasoning for the choice of the thresholds τ_i''s.

5 Experimental results

We run the algorithm over a sequence of 50 scenes from a german highway. Each frame of the sequence is an image with a resolution of 512x256 pixels. An example frame of this sequence in interlaced mode is shown in Fig. 7.

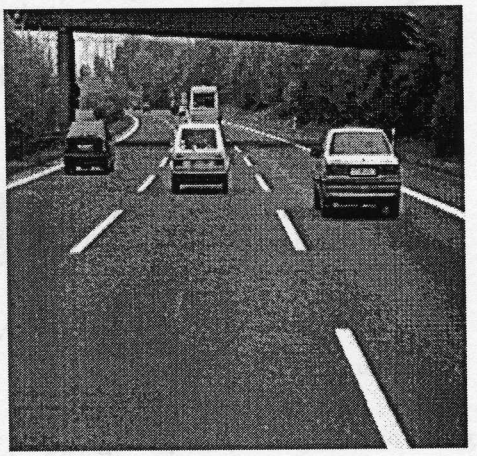

Figure 7: Scene of a german highway

Applying an edge detector to this frame and thresholding we obtain the contour image of Fig. 8. These operations are carried out on a special hardware and can be performed in video rate. For each frame we compute the distance function of the image, which contains at any location in the image the distance to the nearest point.

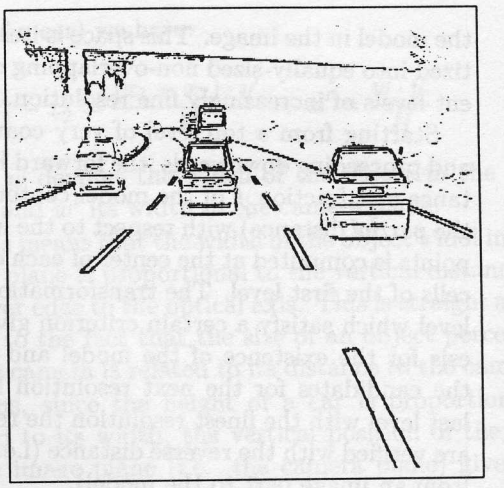

Figure 8: The segmented image

As already mentioned, we use one model for the whole category of cars. A set of instances of it at different scales is calculated in advance and provided as input to the program. In fact only some principal scales could be able to cover the scale dimension, depending on the tolerance of the possible distances. The distance function of the model, which will be used for the reverse matching, is also calculated once in advance for all the scales.

The fraction of the model-points that had to match in the image was kept high (f=80%), and the distance threshold τ was 2 pixels. The algorithm is able to detect the cars of our example frame at the correct positions and scales (see Fig. 9).

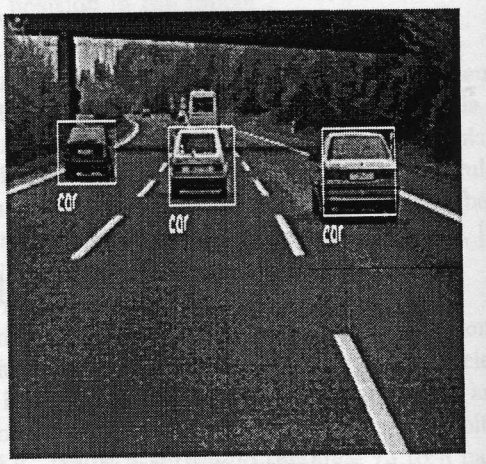

Figure 9: Detected cars

The current implementation of the algorithm is

done in IDL (a high level programming language) and only parts of it are ported into C. Furthermore, the evaluation of the forward criterion at each level of the resolution pyramid can be performed parallely (i.e. for all the cells of a certain level simultaneously), which has not been considered in our case. We can demonstrate the goodness of the method by giving the time needed for one match, which is about 100 μsecs on a Sun SPARCstation 20 (this is the time needed for the calculation of the Hausdorff distance at one position).

6 Summary and Conclusions

We have presented an improvement to the algorithm proposed in [6] which recognizes objects in binary images using the partial Hausdorff distance. This distance transform shows off robustness in the cases of noisy data or partial occlusions. However, in our approach we considered rigid objects.

We use model and geometry information to reduce the dimensionality of the search space only in the translation space. Thus, our algorithm can be considered in our case as more efficient than the primary one.

Problems may arise when the cars are in a far distance from the camera (e.g. near the vanishing point). For such small resolutions the quality of the features extracted from the hardware is poor. We think to face up this case by using different models for small resolutions (i.e. far distances).

The knick of the camera is not considered to be a problem in our case, since the vanishing point estimation is performed for each frame. A further stabilization over the time (i.e time-integration) is expected to increase the performance of the recognition process and could lead to the construction of a robust tracking scheme.

As a further step, we will use the algorithm for the recognition of trucks, since they have considerable differences in the shape and their dimensions compared to cars. Nevertheless, the approach is general and can be applied to any problem of model-based object recognition.

Acknowledgements

The author wish to thank Thomas Kalinke for proof-reading this paper.

References

[1] D.W. Paglieroni, "Distance Transforms: Properties and Machine Vision Applications," *CVGIP*, Vol. 54, No. 1, pp. 56-74, January 1992.

[2] G. Borgefors, "Distance Transforms in Digital Images," *IEEE Trans. on PAMI*, Vol. 8, pp. 344-371, 1986.

[3] P.E. Danielsson, "Euclidean distance mapping," *Comput. Graphics Image Processing*, Vol. 14, pp. 227-248, 1980.

[4] F.P. Preparata and M.I. Shamos, *Computational Geometry, An Introduction*, Springer-Verlag, NY, 1985.

[5] D.P. Huttenlocher,G.A. Klanderman and W.J.Rucklidge, "Comparing Images Using the Hausdorff Distance," *IEEE Trans. on PAMI*, Vol. 15, No. 9, pp. 850-863, September 1993.

[6] D.P.Huttenlocher and W.J. Rucklidge, "A Multi-Resolution Technique for Comparing Images Using the Hausdorff Distance," *in Proc. CVPR*, pp. 705-706, NY, 1993.

[7] D.P. Huttenlocher, J.J. Noh and W.J. Rucklidge, "Tracking Non-Rigid Objects in Complex Scenes," *in Proc. ICCV*, pp. 93-101, 1993.

[8] D.P. Huttenlocher, K. Kedem, M. Sharir, "The upper envelope of Voronoi surfaces and its applications," *in Proc. 7th ACM Symp. Computat. Geometry*, pp. 194-293, 1991.

[9] A.K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, USA, 1989.

[10] T. Kalinke and C. Tzomakas, "Generierung von Objekthypothesen in Verkehrsszenen unter Ausnutzung der Kamerageometrie," IRINI 97-07, Institut für Neuroinformatik, Ruhr-Universität Bochum, Germany.

[11] C. Goerick, D. Noll, M. Werner, "Artificial neural networks in real-time car detection and tracking applications," *Pattern Recognition Letters*, 17, pp. 335-343, 1996.

[12] M. Cappello, M. Campani, A. Succi, "Detection of Lane Boundaries, Intersections and Obstacles," *Intelligent Vehicles 94*, pp. 284-289, 1994.

[13] A. Broggi, "A Massively Parallel Approach to Real-Time Vision-Based Road Markings Detection," *Intelligent Vehicles 95*, pp. 84-89, 1995.