More Local Structure Information for Make-Model Recognition

David Anthony Torres
Dept. of Computer Science
The University of California at San Diego
La Jolla, CA 92093

Abstract

An object classification technique is proposed to solve a vehicle make and model recognition task. Edges of the back end of vehicles are extracted from images. These edges are processed into line segments which contain more local structure information than interest point based characterization can encode. Object matching is performed by comparing the sets of line segments by a Hausdorff distance. The method is tested on a database of vehicle images [2].

1. Introduction

An object classification technique is proposed to solve a vehicle make and model recognition task (MMR). This extends the work done by Dlagnekov [1]. Current MMR is done by extracting scale invariant feature transform (SIFT) features [6] from the image of an automobile. These features are the foundations upon which recognition is performed.

While interest point based feature detectors such as SIFT contribute to robust recognition systems, they focus on minute portions of an image. If an image contains discriminating global features, such as shapes or patterns, the interest point detectors are unable to assimilate them. Vehicles contain such global features by virtue of their contours. Consider the back end of car, it is plausible to suggest that MMR could be performed based on the patterns formed by the tail lights alone. In this paper we explore such an hypothesis by replacing SIFT features with features characterizing contour lines.

Specifically, we take an *edge map*, the binary output of your favorite edge detector, then generate a set of line segments that fit the edge pixels in the map. These line segments are partially invariant to illumination changes and they contain local structure information at a scale that is not available to interest point methods.

The sets of line segments are compared using a line segment Hausdorff distance developed by Gao and Leung in [3]. In general, the Hausdorff distance from a set A to a set B is defined as the maximum distance from a point in A to the nearest point in B, or, formally, $\max_{a\in A}\min_{b\in B} d(a,b)$ where $d(\cdot)$ is some metric[8]. Using line segments as the basis for the Hausdorff distance calculations, Gao and Leung applied the technique to a face recognition task and achieved competitive recognition rates which remained relatively stable in light of illumination, scale, and pose changes. At the heart of this paper is the application of the line segment Hausdorff distance to this noisier make-model recognition task.

The rest of this paper is organized as follows: Section 2 presents the line segment generator. This explains how line segments are obtained from the edge map of an automobile image. Section 3 explains the line segment Hausdorff distance in detail. Section 4 presents results and section 5 concludes.

2. Line segment generator

Line segment generation can be thought of as multiple instances of line fitting a plane curve. The goal of line fitting is to approximate an arbitrary curve by a sequence of line segments. In this case, the arbitrary curve is represented as a set of noisy edge pixels corresponding to the edge map of some image. An edge map is the binary output of some edge detector.

One approach to fitting, known as an error tolerance approach [5], attempts to fit lines such that the perpendicular distance from the point (pixel) to the line is within some error tolerance. One can incorporate a number of minimization criteria to such algorithms. For example, one may minimize the number of line segments produced, or minimize a sum of error term.

For this experiment we implement a simple error tolerant line segment generator which pays no special attention to any minimizing criteria. There is little doubt that improving the line segment generator helps the recognition process. However, the robustness of the recognition procedure should lie heavily on the performance of the line segment Hausdorff distance, and not on the accuracy of the generator. For this reason (or excuse, if you would like,) any line segment generator is accepted which produces, qualitatively speaking, visually recognizable results.

The line segment generator employed is a strip based al-

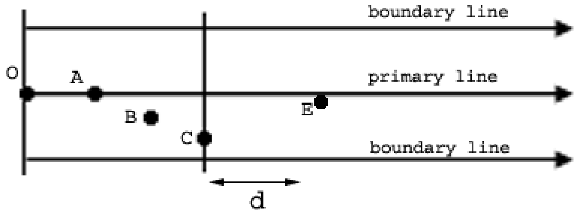

Figure 1: The primary line runs along the direction $\vec{OA}$. Its distance from both boundary lines is d. The end of the strip runs through C because the nearest point (E) is over d away. So the line segment produced is $\overline{OC}$.

gorithm. The idea of using strips to perform line fitting was first proposed by Reumann [7]. A strip is defined by three types of lines as shown in Figure 1, the primary line and two parallel boundary lines which lie a perpendicular distance d from the primary line. The direction of the primary line is determined by the vector between the first two points in the strip $\vec{OA}$. The line segment produced will have end points at the origin O, and at the last point that falls in the strip C. The next line segment is produced by creating another strip with origin at C.

As it is currently defined, a strip extends out to infinity. If this is the case then points that should belong to other logical line segments will fall within the strip and the generator will output an erroneous line. So we must also define the end of the strip as being that position past which there exists no strip point within a parallel distance of d. See Figure 1.

The following process generates all line segments for an image edge map:

1. Find the closest edge point X to the image origin $[0, 0]$.

2. Produce a line segment, $\vec{XC}$, by the strip algorithm using a strip-origin at X.

3. If the previous step fails to produce a line segment (i.e. the strip end encloses no edge points) we arbitrarily elect a new strip-origin and repeat step 2. Otherwise we repeat step 2, taking the next strip-origin at C. This is continued until no new line segments are generated.

Figure 2 shows an image of the back of an automobile, its edge map, and the line segments generated from the edge map.

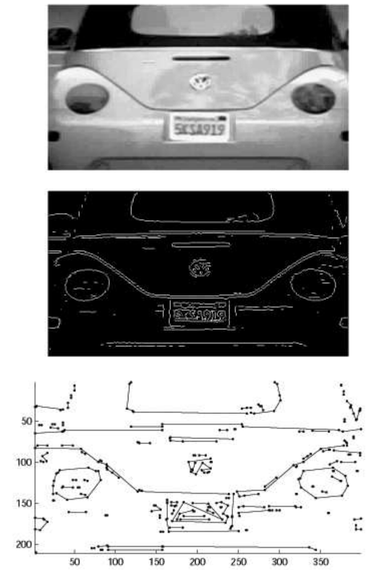

Figure 2: An image, its edge map, and the generated line segments.

3. The line segment Hausdorff distance

The line segment Hausdorff distance (LHD) is a metric which measures the similarity between two sets of line segments. Its output is a non-negative real number such that values close to zero express similarity. The idea behind this work and prior work using LHD, is that one can use the LHD measure to perform object recognition by feeding it sets of line segments which represent the structure of an object's image. In this section we go through an initial description of the line segment Hausdorff distance, this captures the major steps involved in calculating the distance. Then we see how to bolster the LHD heuristically to better deal with errors that occur when the LHD is used in an object recognition task. The LHD described here is developed in [3].

Suppose we have two sets of line segments, a model $M = (m_1, m_2, ..., m_n)$ and input $T = (t_1, t_2, ..., tm)$. The LHD is built upon a vector $\mathbf{d}(m, t)$ which represents a distance, or similarity, between two individual line segments m and t from the sets M and T. This vector is defined as

$$\mathbf{d}(m,t) = \begin{bmatrix} d_\theta(m,t) \\ d_\parallel(m,t) \\ d_\perp(m,t) \end{bmatrix}.$$

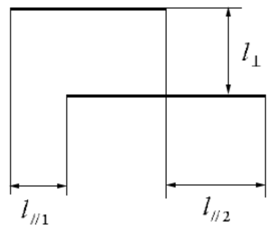

Figure 3: A geometrical view of $d_{\|}$ and $d_{\perp}$.

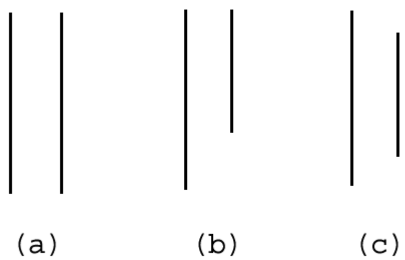

Figure 4: The three cases where $d_{\|}$ is 0.

If we extend the segments m and t so they intersect, $d_\theta(m,t)$ is a function of θ, the smallest intersecting angle between the two lines. Specifically, $d_\theta(m,t) = \theta^2/W$, where the weight W is determined in a training phase of the experiment. d_θ has the property that it penalizes a pair of line segments non-linearly, that is, large intersecting angles result in considerably higher (worse) scores than small angles.

$d_{\|}(m,t)$ and $d_\perp(m,t)$ are demonstrated by considering a pair of parallel line segments as depicted in Figure 3. $d_{\|}(m,t)$, the *parallel distance* between two segments, is the minimum distance that one must translate a segment, in the parallel direction, in order to align either the left or right end points. In Figure 3, this value would be $min\{l_{\|1}, l_{\|2}\}$.

$d_\perp(m,t)$, the *perpendicular distance*, is the distance that one must translate a segment, perpendicular to the parallel direction, to overlap the two segments. In Figure 3 this would be the value $l_\perp$.

Of course, in general, the two line segments will not be parallel. So before calculating the values of $d_{\|}(m,t)$ and $d_\perp(m,t)$, the shortest of the two segments is rotated about its midpoint so that both segments are parallel. The shortest segment is selected because this causes less distortion to the original line segment pair.

Finally, the distance measure returned between the two lines segments is defined as the length of the vector $\mathbf{d}(m,t)$,

$$d(m,t) = \sqrt{d_\theta^2(m,t) + d_{\|}^2(m,t) + d_\perp^2(m,t)}.$$

Now we can define similarity between two *sets* of line segments S and T, this is the line segment Hausdorff distance,

$$LHD(M,T) = max\left(h(M,T), h(T,M)\right),$$

where

$$h(M,T) = \frac{1}{\sum_{m\in M}|m|}\sum_{m\in M}|m|\cdot\min_{t\in T} d(m,t),$$

and $|m|$ denotes the length of line segment m.

The general definition of the LHD is complete. Now we augment the definition to help alleviate errors that may occur when we use the LHD in an object recognition task. First, we deal with the effect of broken lines due to segmentation error. In other words, a line segment m in some model set is broken, due to noise, into several smaller segments $\{t_1, t_2, ...t_n\}$ in the input set. To deal with this, $d_{\|}(m,t)$ returns zero whenever the range of one line segment is completely contained in the other. Figure 4 shows the three such cases when this occurs.

A benefit of calculating the LHD is that it never explicitly matches pairs of line segments. But in obviating this expensive procedure, it allows for another type of error to occur. Consider a model set of line segments M and a corresponding input set T. Say $m \in M$ has no corresponding segment in T due to noise, and, some incorrect input set N by chance contains a segment that correlates better with m. Several such errors may add up so that $h(M,N)$ gives a better score than $h(M,T)$, even though the latter is the correct correspondence.

To help curb this type of error, a mechanism is added to the LHD that heuristically tracks how many line segments will probably be correctly matched. The heuristic introduced comes to us in the form of a similarity neighborhood: a combination of a positional neighborhood N_p and an angular neighborhood N_a. If a test segment's intersecting angle is within N_a then the segment falls within the angular neighborhood. If each of the two end points of a test segment fall within a radius of N_p of the model end points, then the segment falls within the positional neighborhood.

The rationale behind the similarity neighborhood is if one or more input line segments fall within the positional and angular neighborhoods of a model segment m, then m will likely be correctly matched. m is called a *high confidence line*. We define the *confidence ratio* R between a model set and test set as the ratio of high confidence lines (N_{hc})to the total number of lines in the model set (N_{total}).

$$R = \frac{N_{hc}}{N_{total}}$$

This new information and the existing LDH are integrated into a new Hausdorff distance measure

$$LHD'(M,T) = \sqrt{LHD^2(M,T) + (W_n D_n)^2},$$

where W_n is a weight to be determined in a training process, and D_n is the *number disparity* defined as the average ratio of lines found outside of the similarity neighborhood of the two sets to be compared,

$$D_n = 1 - \frac{R_M + R_T}{2} = \frac{(1 - R_M) + (1 - R_T)}{2}.$$

R_M and R_T are the confidence ratios of the model and test sets, respectively.

Four parameters must be tuned before LHD' is used for testing. These are W, W_n, N_p, and N_a. Simulated annealing [9] [4] is used to approximate the parameter values that lead to a global minimization of the error rate on a set of test data.

4. Results

The make-model recognition task is evaluated using a database of vehicles images [2]. The database consists of 1103 cropped images of the back end of vehicles of various makers and models. Also available is a set of 38 query images used to test the make and model recognition algorithm.

Direct comparison of a query image to each of the 1103 images in the test database is infeasible due to the computational complexity of the LHD. In order to obtain results in reasonable time the test database is dramatically reduced by selecting one image corresponding to each make-model class in the query data set.

Note that the license plate portion of the images is cropped out to eliminate any noise that the region may contribute to the LHD.

Table 1:
Recognition Rate of LHD

Recognition Rate	0.4211
Top3 Rate	0.6579
Top5 Rate	0.7368

The experimental results are summarized in Table 1. The LHD is able to achieve a recognition rate of 0.42. The Top3/Top5 Rate is defined as the proportion of times that the corresponding model image is found in the top 3/5 matches of the query results.

Figure 5: Sample cropped images used in experiment 2.

Table 2:
Recognition Rate for Cropped Data Set

Recognition Rate	0.5966
Top3 Rate	0.7524
Top5 Rate	0.8264

Another set of experiments is run in which the images are cropped so that only the bumper portion of the image is visible (Figure 5). This is done to remove any noise which may occur due to the image background or to windshield reflection and specularity. Table 2 shows that the recognition rates are markedly better.

5. Discussion

The proposed make-model recognition system achieves good recognition rates, though the performance is not nearly as good as the SIFT based system described by Dlagnekov [1]. We can attribute some of this performance difference to the computational complexity of the LHD. It takes significantly longer to obtain recognition result via LHD. As a consequence, only a limited number of images can be used as a model database which affects the overall recognition rate of the system. Also, because of computational complexity and practical time constraints, the simulated annealing step can only be run for a limited number of time steps, thus, is it difficult to say if near optimal parameter values are being estimated.

The LHD was coded in this project using Matlab, so a definite speed gain is expected by implementing the algorithm in a low level language such as C/C++. Also, as Gao and Leung report [3], there is a fair amount of speed-up to be gained by implementing *prefiltering techniques* which are techniques that attempt to discard obvious non-matches before the costly LHD is calculated. Speed-up, which allows for more database comparisons and allows for further parameter exploration is beleived to be key in producing a competetive results.

6. Conclusion

A vehicle make-model recognition system is presented. In the first stage of the system, edges from images of the back end of vehicles are extracted using a standard edge detector. From the resulting edge map we extract a set of line segments using a line fitting algorithm. The line segments capture local structure information. One compares two sets of line segments using a line segment Hausdorff distance. Classification of an input set is based on a minimal Hausdorff distance between the input set and its matching class.

Recognition rates for the system are up to 59.7%, which is good, though SIFT based recognition systems can obtain classification rates of 89.5% [1]. The gap in success rates is due in part to the computational complexity of the LHD which forces the experiment performed here to be far simpler and less realistic, so a straight comparison of both methods may not appropriate.

References

[1] L. Dlagnekov. Video-based car surveillance: License plate make and model recognition. Masters Thesis, University of California at San Diego, 2005.

[2] L. Dlagnekov and S. Belongie. Ucsd/calit2 car license plate make and model database. http://vision.ucsd.edu/car_data.html, 2005.

[3] Y. Gao and M. K. H. Leung. Face recognition using line edge map. In *IEEE Trans. on Pattern Analysis and Machine Intelligence Vol. 24, pp.764-779*, 2002.

[4] R. O. Duda P. E. Hart and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., New York, second edition edition, 2004.

[5] M. K. H. Leung and Y. H. Yang. Dynamic two-strip algorithm in curve fitting. In *Pattern Recognition, vol. 23, pp. 69-79*, 1990.

[6] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[7] K. Reumann and A. P. Witkam. Optimizing curve segmentation in computer graphics. *Int. Comput. Symp.*, pages 244–256, 1974.

[8] G. Rote. Computing the minimum hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, 38:123–127, 1992.

[9] P. J. M. van Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, New York, 1987.