A Tracker
for Broken and Closely-Spaced Lines

Naoki Chiba **Takeo Kanade**

October 1997
CMU-CS-97-182

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We propose an automatic line tracking method which can deal with broken or closely-spaced line segments more accurately than previous methods over an image sequence. The method uses both grey scale information of the original images and geometric attributes of line segments.
By using our hierarchical optical flow technique, we can get a good prediction of line segments in a consecutive frame even with large motion. The line attribute of direction, not the orientation, discriminates closely-spaced line segments because when lines are crowded or closely-spaced, their directions are opposite in many cases, even though their orientations are the same. A proposed new matching cost function enables us to deal with multiple collinear line segment matching easily instead of using one-to-one matching. Experiments using real image sequences taken by a hand-held camcorder show that our method is robust against line extraction problems, closely-spaced lines, and large motion.

This research is partially sponsored by the Advanced Research Projects Agency under the Department of the Army, Army Research Office under grant number DAAH04-94-G-0006. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the Department of the Army, or the U. S. government.

Keywords: Computer vision, Motion Estimation, Optical flow, Line tracking

1 Introduction

Automatic line segment tracking through an image sequence is one of the difficult problems in motion estimation due to some reasons. First, due to the difficulty of line extraction, the extracted end points of each line segment are not reliable. Furthermore, a single line segment tends to be broken into multiple line segments over image frames. Second, when line segments are very closely-spaced, it is hard to track and distinguish one line segment from another because they have similar orientations.

The procedure for the line tracking method consists of two steps: a prediction step and a matching step. For the prediction step, there are two types of approaches: prediction by a Kalman filter [4] and prediction by the epipolar constraint between images [1, 10]. The problem of the Kalman filter-based techniques is that it has difficulty in setting up the uncertainties for the segment tracking which is usually tuned by hand. Although the epipolar constraint is geometrically powerful, it requires a good method to obtain from unknown motion which is usually sensitive to noise. For the matching step, several attributes of each line segment have been introduced [1, 6, 3]. They include the end-point and the orientaion of each line, the distance and the overlapping length between line segments. However, their methods are easily confused when the input consits of closely-spaced line segments, and they cannot deal with broken line segments.

We propose a new line tracking method which has a reliable prediction based on hierarchical optical flow for large motions, an accurate line attribute of the directions, and a robust matching function to track multiple collinear line segments. For the optical flow estimation, various kinds of techniques have been investigated by many researchers. However, there remains one big problem: these techniques cannot deal with insufficiently textured areas. We solve the problem by using a simple filling operation. Even if a line prediction is precise, it is hard to distinguish closely-spaced lines. We introduce a line direction attribute extracted at the edge extraction stage. Many matching functions have already introduced. However, very few of them deal with multiple line segments. We propose a simple and expandable matching function which can deal with collinear multiple line segments.

The method is robust enough to deal with real image sequences taken by a hand-held camcorder. It can be used to provide line correspondences over an image sequence for recovering the shape of a man-made object with line features, such as buildings or indoor scene of a room.

In the following sections, line segments are extracted from grey scale images by using the Vista Image Processing package. The edges are extracted by the Canny operator with hysteresis thresholding. The edgels are then linked to a set of edges and segmented into straight lines by using Lowe's method [7].

2 Optical Flow Prediction

2.1 Lucas-Kanade Method

The Lucas-Kanade method is one of the best optical flow estimation techniques, because it is fast, simple to implement and controllable because of the tracking confidence [8, 2]. This method assumes that images taken at near time instants are usually strongly related to each other, because they refer to the same scene taken from only slightly different viewpoints. In general, any function of three variables $I(x, y, t)$, where the space variables x and y as well as the time variable t are discrete and suitably bounded, can represent an image sequence. Using this assumption, the function $I(x, y, t)$ satisfies the following property:

$$I(x, y, t) = I(x + \xi, y + \eta, t + \tau) \tag{1}$$

This means that a later image taken at time $t + \tau$ can be obtained by moving every point in the current image, taken at time t, by a suitable amount. It assumes that the intensity of each point is constant between frames. The amount of motion $\mathbf{d} = (\xi, \eta)$ is called the displacement of the point at $\mathbf{u} = (x, y)$ between time instants t and $t + \tau$. Our goal is to find the displacement vector $\mathbf{d}$ of a point from one frame to the next.

An important problem here is that a single pixel cannot be tracked, because the equation gives us only one constraint to solve for the two unknown parameters ξ and η. The Lucas-Kanade method assumes that neighboring pixels in a small window have the same flow vectors. Then we can choose the displacement vector $\mathbf{d}$ so as to minimize the residue error defined by the following equation, if we redefine $J(u) = I(x, y, t)$ and $I(u + d) = I(x + \xi, y + \eta, t + \tau)$

$$E = \sum_w [I(u + d) - J(u)]^2 \tag{2}$$

By using the linear approximation in the form,

$$I(u + d) = I(u) + I^{'}(u)d \tag{3}$$

The minimum error occurs when the derivative of E with respect to d is zero.

$$\begin{aligned} 0 &= \frac{\partial}{\partial d} E \\ &\approx \frac{\partial}{\partial d} \sum_w [I(u) + dI^{'}(u) - J(u)]^2 \\ &= \sum_w 2I^{'}(u)[I(u) + dI^{'}(u) - J(u)] \end{aligned} \tag{4}$$

from which

$$d \approx \frac{\sum_w I^{'}(u)[J(u) - I(u)]}{\sum_w I^{'}(u)^2} \tag{5}$$

Having obtained this estimate, we can then move I(x) by our estimate of d, and repeat this procedure, yielding a type of Newton-Raphson iteration. This iteration is expressed by the following equations.

$$d_0 = 0 \tag{6}$$

$$d_{k+1} = d_k + \frac{\sum I^{'}(u + d_k)[J(u) - I(u + d_k)]}{\sum I^{'}(u + d_k)^2} \tag{7}$$

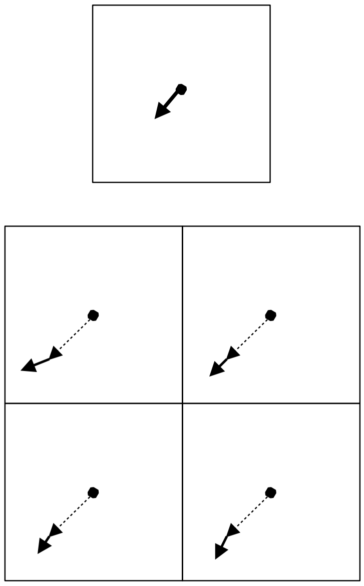


Figure 1: Coarse-to-fine

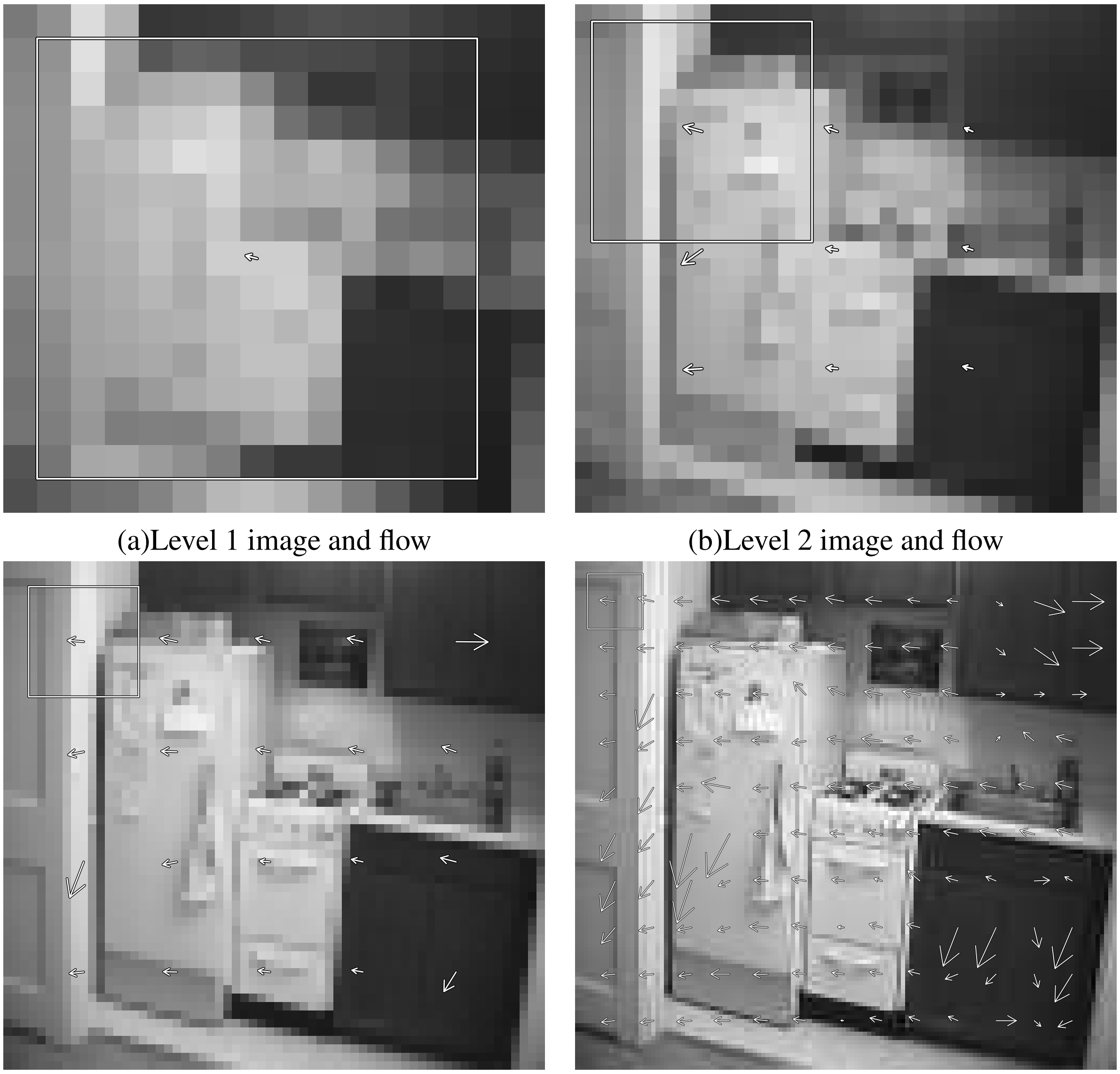
2.2 Coarse-to-fine Multi-resolution

The biggest drawback of gradient methods, including the Lucas-Kanade method, is that they cannot deal with large motion between frames because they use a linear approximation. Coarse to fine multi-resolution techniques could be applied to solve this problem. Although the original Lucas-Kanade method for the stereo registration problem was interactive, coarse-to-fine multi-resolution techniques could make the method completely automatic for optical flow estimation.

Before tracking, we prepare a set of multi-resolution images like a pyramid. Suppose we have two levels of image like Figure 1, the coarser level has a smaller image size. We partition the image into regions to compute optical flow. In each region we assume that all pixels have the same optical flow vectors. The rectangles indicate the regions. Flow results at the coarser level (Level i) will provide initial estimates as starting values for tracking in the next finer level (Level i+1) of the pyramid. The dotted arrows indicate flow results at the previous level, which are scaled to the current image size. At the current level (Level i+1), we start tracking from those points to get optical flow. Solid arrows indicate computed flow at this level. The summation of the previous level's results (dotted arrows) and current level's results (solid arrows) will be the actual optical flow vectors at this level.

A big problem here is that many regions of typical real images do not contain sufficient texture to produce reliable optical flow estimates. Furthermore, in the process of making multi-resolution images, regions that have tiny features surrounded by textureless regions tend to be smoothed over. Figure 2 shows a result which contains insufficient texture. While the flow results are correct in highly textured regions, the flow results are very poor in regions containing insufficeint texture. The box at the upper-left corner of each image indicates the size of the region used for flow computation.

Poelman solved this problem to some extent by using a confidence-weighted smoothing method



(a)Level 1 image and flow

(b)Level 2 image and flow

(c)Level 3 image and flow

(d)Level 4 image and flow

Figure 2: Optical Flow with textureless regions

between the levels of the pyramid in [9]. The overview is as follows. One of the advantages of the Lucas-Kanade method is that we can measure its tracking confidence. Tomasi and Kanade introduced a method for obtaining a region's trackability from the image derivatives in [11]. By examining the two eigenvalues of the following 2×2 coefficient matrix G, we can determine whether a region of the image consists of a *trackable* feature point.

$$G = \sum_{w} \left(\frac{\partial I}{\partial x}\right) \left(\frac{\partial I}{\partial x}\right)^T \tag{8}$$

Relatively textureless regions are difficult to track and have very low image derivatives. Therefore their corresponding G matrices have smaller magnitudes and intensity variation. However simply examining the magnitudes of the G matrices is not sufficient for determining trackability. Regions containing high spatial derivatives can be line or edge feature, which can only be localized in one direction. If the horizontal and vertical gradients are highly correlated, as in the case of oriented features, the G matrix will have one big eigenvalue and one small eigenvalue. When both eigenvalues are large, the region has high spatial gradient in two orthogonal directions, therefore it can be localized well in the image. Thus we can measure a region's trackability by the smaller eigenvalue.

From the smaller eigenvalue λ_{min} and the tracking residue E, we can obtain a tracking confidence γ by the following equation.

$$\gamma = \frac{\lambda_{min}}{E} \tag{9}$$

By using this confidence to solve the problem of lack of texture, Poelman weighted inter-level results by their confidences. The method uses the result of the lower resolution level, not only as an initial estimate for iterative tracking at the next level but also as a weighting factor to combine the current level flow estimate and the previous level flow estimate.

However, the result is still not good enough for some reasons. First, the behavior of the confidence value does not vary linearly with the amount of texture, especially when the value is small, i.e. the region does not have sufficient texture. When the region's amount of texture is too little, we cannot trust the estimate at all. Second, when a region contains a tiny feature at the highest resolution and is surrounded by regions that have insufficient texture, the feature is smoothed over at most of the coarser levels. The inherited flow results from the coarser levels tend to contain significant errors before reaching the finest level. This is because it uses only the upper level's results.

2.3 Filling by Dilation

We propose a filling method after thresholding at an intra-level, rather than smoothing or weighting. We use the estimate at the lower resolution level as an initial value only for the iterative flow estimate at the current level. After estimating the optical flow fields, we threshold the fields with their confidences, and discard poor estimates. We do not trust the results any more if the region's confidences are smaller than a predefined amount. We assume that regions that have insufficient texture have similar flow results to their neighboring regions which have reliable estimates. Analogously to the dilation technique for filling holes in binary images, unreliable optical flow fields can be filled by using dilation technique as well.

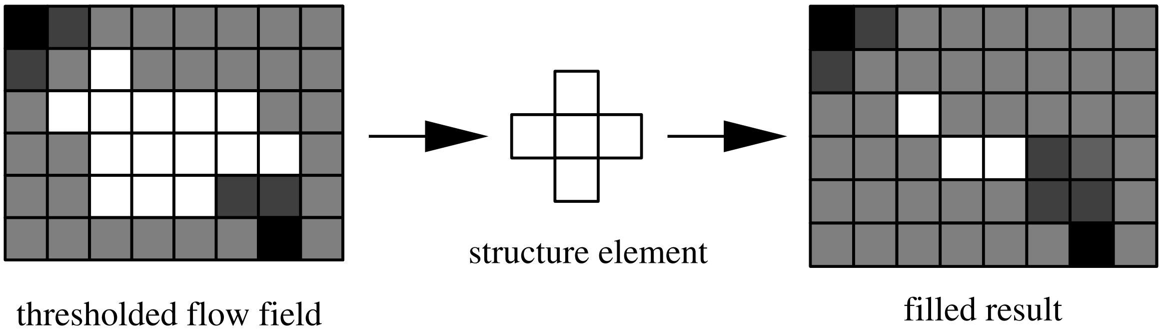


Figure 3: Dilation

Figure 3 illustrates the operation of flow vector dilation. Dark pixels -the darker is better- represent reliable flow results which have high confidences. White pixels represent below-thresholded flow results. We can fill the blank fields by using the dilation operation. A flow vector, $u(i,j)$, at image coordinates, i, j, is calculated by its four neighboring values weighted by their confidences, γ. The operation continues until all fields are filled with the proper estimate.

$$\begin{aligned} u(i,j) &= \sum_{p,q \in w=\{-1,1\}} \frac{\gamma(i+p,j+q)u(i+p,j+q)}{\gamma_A} \\ where & \\ \gamma_A(i,j) &= \sum_{p,q \in w=\{-1,1\}} \gamma(i+p,j+q) \end{aligned} \tag{10}$$

Figure 4 shows thresholded flow results in each level of image. X marks indicate that the regions have poor results and they have thresholded out. Unreliable flow results are discarded, only flow results in regions that have sufficient texture remain.

Figure 5 shows filled flow results in each level of image. In regions containing little or no image texture, the flow results have been filled by results of surrounding highly textured regions.

By using this filling based hierarchical optical flow estimation, we can obtain a good prediction for automatic line segment matching between a pair of images. One good thing of our method is that we do not have to do anything other than setting the threshold value. The value depends on images, but it is not difficult to find an appropriate value. In our experiments, we used 100 as the value.

2.4 Prediction Result

Using a pair of real scene frame images, we compared our method with Poelman's method. Figure 6 shows the whole image of a typical case with textureless regions, in which we need reliable optical flow to track line segments. While the upper right part around the kitchen cabinet does not have sufficient texture, there is a line between the cabinet doors. Therefore we need reliable optical flow even in that region.

To compare the methods, we generated the second frame images from the intensity image in the first frame and dense optical flow between the frames estimated by each of the methods. Figure

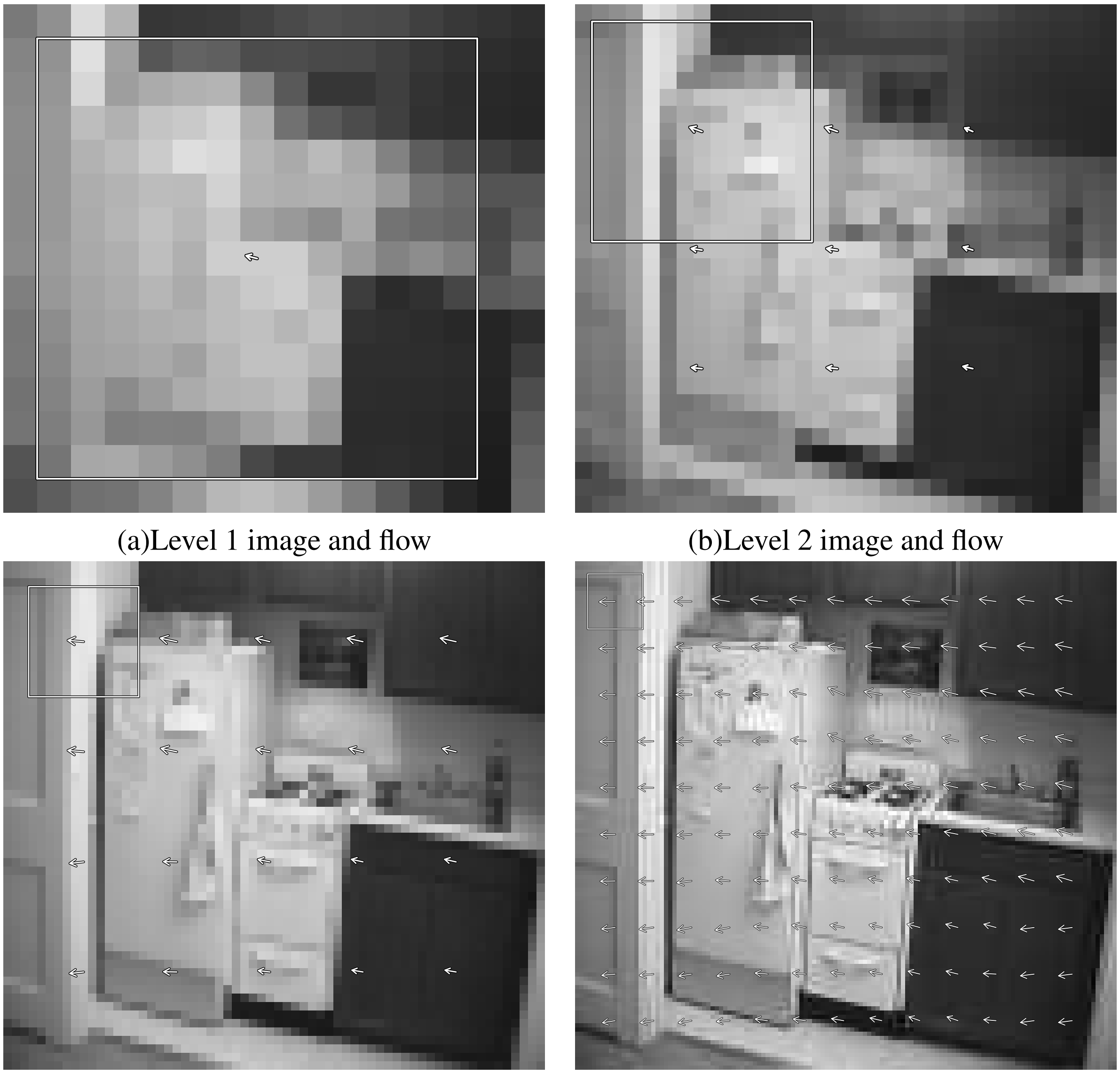
(a)Level 1 image and flow

(b)Level 2 image and flow

(c)Level 3 image and flow

(d)Level 4 image and flow

Figure 4: Thresholded Optical Flow



(a)Level 1 image and flow

(b)Level 2 image and flow

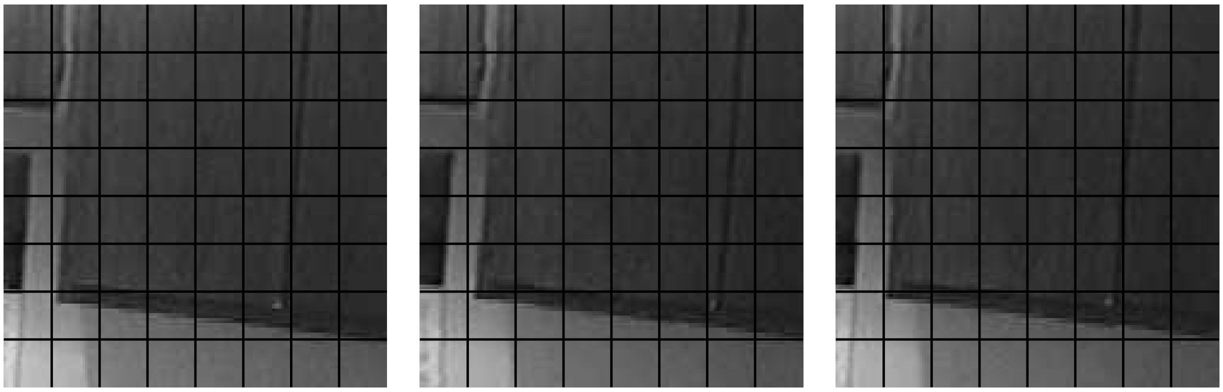
(c)Level 3 image and flow

(d)Level 4 image and flow

Figure 5: Optical flow using dilation

Figure 6: Kitchen Sequence

7 shows the upper right part of the true second image and the generated images, overlayed with grid lines to show the difference. The grid size corresponds to 20 pixels. When we look at a line between the cabinet doors in the true image, it is in both the second and third columns from the right. However, in the image generated by Poelman's method, it has shifted to the right, therefore the line is in only the second column from the right with more than 5 pixel errors. On the other hand, we can see that the image by our method is much more similar to the true image in all of the areas including the line within one or two pixel errors.

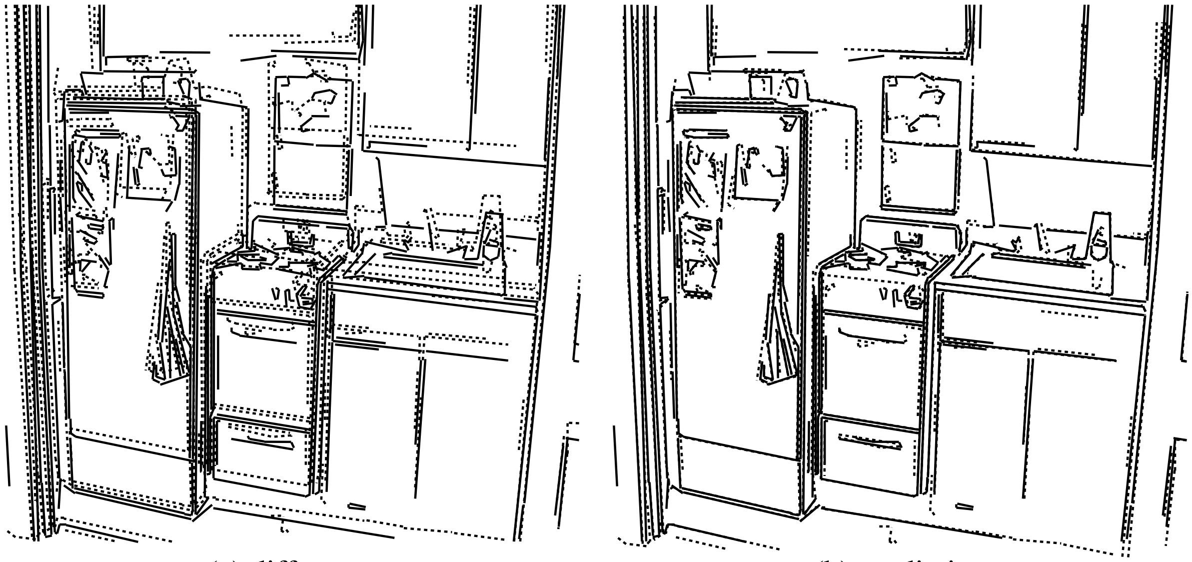


(a)true (b) Poelman's (c) our method

Figure 7: The Prediction Result

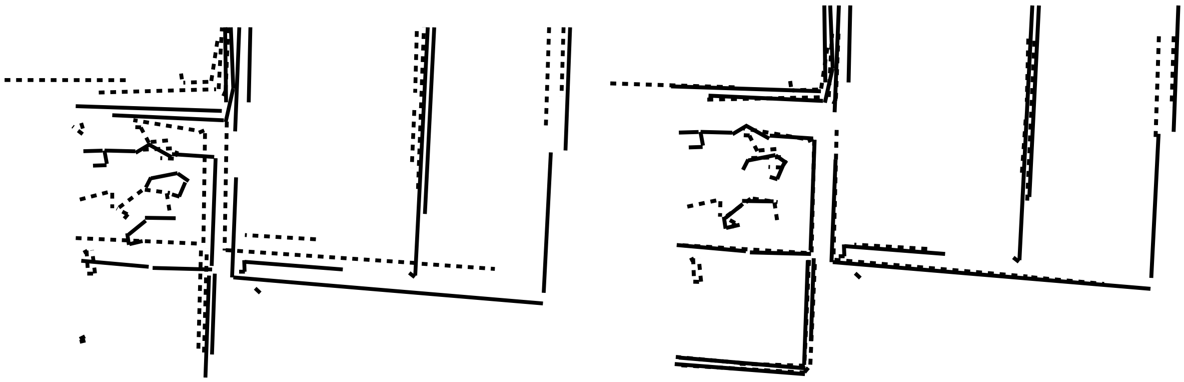
By using the optical flow estimation technique described above, we can predict line segments over images. Figure 8 (a) shows the position difference of line segments due to camera motion from one frame to the next. The solid lines represent line segments at the current frame and the dotted lines represent line segments at the previous frame. We can see that there is some amount of motion between the frames. Figure 8 (b) shows the predicted line segments by our improved optical flow estimation technique. The dotted lines represent predicted line segments from the previous frame using our optical flow estimate. The solid lines represent observed line segments at the current frame.

Most of the line segments are properly predicted with one or two pixel distance errors, although their lengths, in other words the positions of end-points along the lines, have larger errors due to the line extraction problem. In order to have a close look at the prediction, figure 9 shows the detail of the frame difference and the prediction of the upper part area. We can see the efficiency of our optical flow estimation even in regions that do not have sufficient texure.



(a) difference (b) prediction

Figure 8: Prediction of Line Segments



(a) difference (b) prediction

Figure 9: Detail of the prediction

3 Matching Function

3.1 Line Direction Attribute

Various attributes of each line segment have already been introduced for tracking lines or matching lines to the object model. Many of them use attributes derived only from end points. For example, Crowley et al used the following attributes [3] .

θ orientation

h half length

x_c horizontal coordinate of the center point

y_c vertical coordinate of the center point

c parpendicular distance from the origin

a, b line equation coefficients

However, these geometric attributes are not good enough to discriminate line segments, when they are crowded, parallel and closely-spaced to each other.

On the other hand, we could use more information from the grey level images. For example, Schmid and Zisserman used intensity neighborhoods of the line as grey level information [10]. Still, it is not good at dealing with closely-spaced line segments. Because it requires a proper size of square neighborhood to cover only the line's neighbourhoods. It also assumed that the correct epipolar constraint between two images is provided for getting the point to point correspondance for the cross-correlation. Epipolar geometry is not easy to be determined from unknown motion with an uncalibrated camera such as images taken by a hand-held camcorder. An edge profile is a curve showing how the intensity changes across an edge. Kanade[5] showed that an edge profile is a good way to find similar line segment. However, setting up a proper length for extracting an edge profile is difficult when lines are very closely-spaced.

We propose another attribute, line direction attribute, not the orientation, which we can obtain from the intensity images. This denotes which side of a line is brighter. The edge extraction algorithm not only gives us edge strengths but also edge directions. Since each edgel already has its own direction, we do not have to extract its edge direction again and do not have to set the length for extraction. All we have to do is to take the average of all edge directions in a line segment.

From the average edge direction of a segment, we can define a line direction attribute of a line segment, which represents not only the orientation but also which side is brighter in the intensity image. We represent the line direction with the order of end point coordinates. For example, we will reorder the end points so that when moving from the starting end point to the ending end point, the right hand side of the line is always brighter.

We found that this line direction attribute works very well for discriminating closely-spaced lines in real image sequences as long as the direction does not change, and is very fast to compute.

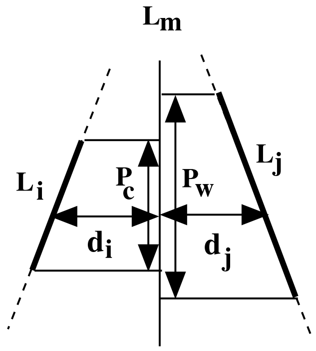


Figure 10: Similarity Function

3.2 Similarity Function

Several similarity functions have already been introduced. They measure the orientation, the distance, and the overlaps between two line segments. However they consider only one-to-one matching. In real images, a single line segment may be broken into multiple line segments very easily.

We propose a similarity fuction which is expandable to matching multiple line segments. For the purpose of measuring the similarity between lines L_i and L_j, let us define the average distance D and the overlapping ratio P. In figure 10, the middle separating line L_m is the axis of symmetry between the two lines. Let d_i denotes the average distance of end points between line L_i and the middle line L_m. Let P_c denotes the length of overlapping part of the projected lines to the middle line Lm, and P_w denotes the length of the whole part.

The distance D from the middle line and the overlapping ratio P, the ratio between the overlapping part and the whole part of the projected lines onto the middle line, are as follows.

$$D = d_i + d_j \tag{11}$$

$$P = \frac{P_w}{P_c} \tag{12}$$

Intuitively, when two lines are similar, the distance D are small and the overlapping ratio P are close to one. Thus considering the line direction, $q(i)$ and $q(j)$, we will define similarity s(i, j) as follows.

$$s(i,j) = \begin{cases} \frac{1}{k_d D + k_p P} & \text{if } q(i) = q(j) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

In the equation, k_d and k_p are coefficient factors determine how much each term affects the similarity. These will be set to be proper values from experiments.

3.3 Multiple Line Matching

One single line segment tends to be broken into multiple segments very easily due to the difficulty of line extraction which is sensitive to lighting condition changes. The proposed matching function is easily expandable to the case of multiple line segments.

Figure 11 illustrates a simple example. When line L_k is collinear to line L_j and the gap is reasonably small, we can consider them as a single combined line segment. Collinear line segments with small gaps are checked and registered beforehand as possible combinations to form a line segment. Figure 12 illustrates the possible matching patterns. In order to find the most similar line for L_i, we can examine not only the single lines L_j and L_k separately, but also the combined line of L_j and L_k.

In this example, the distance between lines are simlilar, but the overlapping ratio is quite different from each other. We will look for the most similar set of line segments from all possible combinations by using the same similarity function.

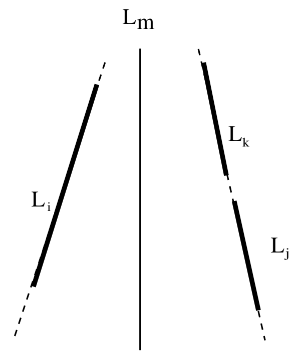
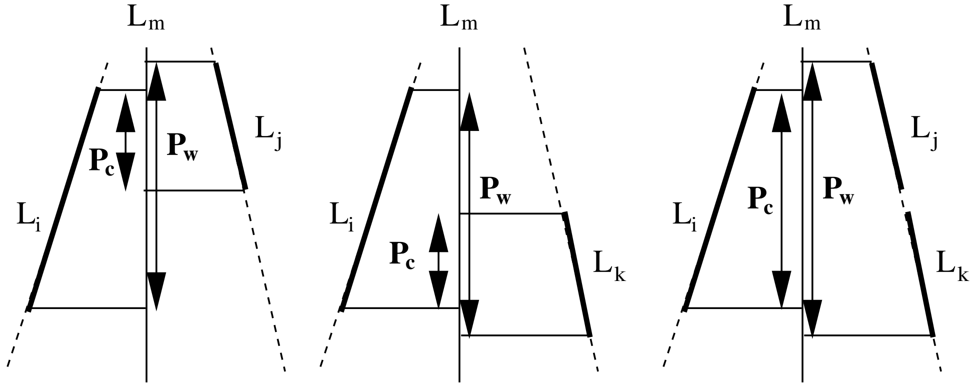


Figure 11: Multiple Matching



(a) L_j (b) L_k (c) combined

Figure 12: Possible Matching Patterns

4 Experiments

In this section, we present the results achieved by applying our line tracker to real image sequences taken by a hand-held camcorder. In the following experiments, we used 0.5 for the k_d and 1.0 for the k_p to match line segments.

(a) selected lines (b) the tracked lines

Figure 13: The Tracked Result (Kitchen)

Figure 14: The Tracked broken line

4.1 Kitchen Sequence

Figure 13 shows the result of kitchen sequence with 10 frames. In the first frame, 384 line segments are obtained using the extraction method described in the introduction. Figure 13 (a) shows 100 of important line segments which are selected automatically according to their edge strengths and line lengths. Figure 13 (b) shows the tracked line segments at the 10th frame. Out of 100 line segments, 93 line segments are correct. 85 of them have been visible over the entire sequence and tracked correctly, 4 of them are properly detected out of field. The disappearance of 4 of them (caused by the line extraction) is properly detected. Most of the wrong tracking is caused by imperfect edge extraction.

(a) First frame (b) Second frame (c) 16th frame

Figure 15: The Tracked Result (Building)

4.2 Broken Lines

Figure 14 shows the tracked broken lines. The line is a single line in the previous frame (left), but it is broken into two lines in the next frame (right) which has a 3 pixel gap. Our method tries to examine all possible combinations to get a better similarity.

All line segments are checked beforehand whether they are collinear and the gaps are small. Collinearities can be checked by the line equations. Collinear line segments which are separated by gaps smaller than a predefined amount will be registered as a possible combination to form a single line segment. In this experiment, we use the gap of 10% of the combined line segment length.

The single line in the left frame compared with the combination of two line segments in the right frame has a better similarity value compared to either of the lines alone, because the overlapping ratio of the combined segment is higher than the separate segments.

4.3 Building Sequence

Figure 15 is another example of the tracking results. This sequence has a greater number of line segments, larger motion, and more frames (16 frames) than the kitchen sequence. Due to the large range motion, there is large deformation between the first frame and the last frame. Because of the depth of the scene, the optical flow vectors are not uniform. Points closer to the camera have larger motion whereas points further away have smaller motion. For example, the closer building has more than 50 pixels of motion, even though the left side tower has around 20 pixels between the first frame and the second frame with an image size of 640 by 480. Although our goal is to track line segments in a static scene with camera motion, this sequence includes moving people and tree shadows. The sequence was taken by a hand-held consumer camcorder and digitized on an SGI Indy. We can say that this is a very noisy sequence.

In the first frame, 610 line segments are extracted and 200 of them are automatically selected to be tracked in the same way as the kitchen sequence. Figure 15 (a) shows the selected line segments. In the second frame, 190 of those line segments are correctly tracked. The tracker could detect all of the existing 12 out of field lines, but only 2 of the 5 lines that have disappearred. Most of the wrongly tracked lines are around the tree shadow at the bottom left or the moving people. In the

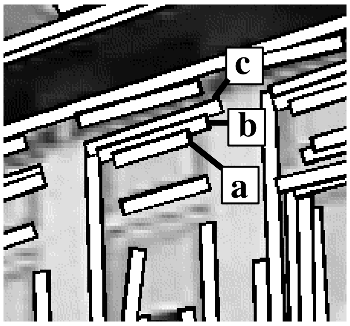


Figure 16: The Tracked Closely-Spaced Lines

last frame, 48 line segments remain and all of them are correctly tracked.

4.4 Closely-Spaced Lines

Figure 16 shows an example of closely-spaced line tracking. This is the upper part of the center window on the second floor of the closer building. Although the line segments a, b, and c are very closely-spaced, have similar lengths and orientations, the directions of a and b, or the directions of b and c, are different. Therefore all three of them are correctly tracked by our method until they went out of field.

The experiments show that very good results are obtained using real image sequences. In the kitchen sequence even lines surrounded by regions that have insufficient texture are correctly tracked by using correct optical flow estimation. In the building sequence, we showed that the tracker is robust enough to discriminate very closely-spaced lines.

5 Conclusion

This paper has addressed the problem of tracking line segments over an image sequence, especially for sequences taken with unknown motion. We have demonstrated that our line segment tracker is very robust for noisy real image sequences. It can track not only simple lines but also broken lines or closely-spaced lines.

The method includes three novel techniques to realize a robust line tracker for real image sequences. First, by using our new coarse-to-fine optical flow estimation technique, we can get a very good prediction of line segments between frames. Second, we showed that the line direction attribute is very helpful for discriminating closely-spaced lines when we match line segments between frames. Third, the proposed similarity function can overcome the imperfection of edge extraction by making it possible to be applied to real image seqeunces containing many broken line segments.

An important feature of the method is that it requires nothing but image sequences, i.e., it does not require a camera calibration or constrained camera motion. Thus we can track line segments in an image sequence taken by a hand-held camcorder for shape recovery.

References

[1] N. Ayache and B. Faverjon, “Efficient registration of stereo images by matching graph descriptions of edge segments”, International Journal of Computer Vision, pp. 107-131, 1987.

[2] J. Baron, D. Fleet, and S. Beauchemin “Performance of Optical Flow Techniques”, International Journal of Computer Vision, pp. 43-77, 1994.

[3] J. Crowley, P. Sterlmaszyk, T. Skordas, and P. Puget “Measurement and Integration of 3-D Structure by Tracking Edge Lines”, International Journal of Computer Vision, Vol. 8, No. 1, pp.29-52, 1992.

[4] R. Deriche and O. Faugeras, “Tracking line segments”, Proc. ECCV’90, pp. 259-268, 1990.

[5] T. Kanade “Recovery of the Three-Dimensional Shape of an Object from a Single View”, Artificial Intelligence, 17, pp. 409-460, 1981.

[6] D. G. Lowe, “The Viewpoint Consistency Constraint”, International Journal of Computer Vision, Vol.1, pp. 55-72,1987.

[7] D. G. Lowe, “Three-dimensional object recognition from single two-dimensional images”, Artificial Intelligence, 31, pp. 355-395, 1987.

[8] B. Lucas and T. Kanade, “An Iterative Image Registration Technique with an Application to Stereo Vision”, DARPA Image Understanding Workshop, pp. 121-130, 1981.

[9] C. Poelman and T. Kanade, “The Paraperspective and Projective Factorization Methods for Recovering Shape and Motion”, Technical Report CMU-CS-95-173, Carnegie Mellon University, 1995

[10] C. Schmid and A. Zisserman, “Automatic Line Matching across Views”, Proc. CVPR’97, pp.666-671, 1997.

[11] C. Tomasi and T. Kanade, “Shape and Motion from Image Streams: a Factorization method - Part 3 Detection and Tracking of Point Features”, Technical Report CMU-CS-91-132, Carnegie Mellon University, 1991