A TRACKER FOR BROKEN AND CLOSELY-SPACED LINES

Naoki CHIBA

Chief Researcher,
Mechatronics Research Center,
SANYO Electric Co., Ltd., JAPAN
chiba@mech.rd.sanyo.co.jp

Takeo KANADE

Professor,
School of Computer Science,
Carnegie Mellon University, USA
tk@cs.cmu.edu

Commission V, Working Group IC V/III

KEY WORDS: Line Tracking, Motion Estimation, Optical Flow, Morphology, Shape Recovery

ABSTRACT

We propose an automatic line tracking method which is robust in tracking broken and closely-spaced line segments over an image sequence. The method uses both the grey-level information of original images and the geometric attributes of line segments.
By using a hierarchical optical-flow estimation technique, we can obtain a good prediction of line segment motion in a consecutive frame. There remains the need to distinguish closely-spaced lines which are common in man-made objects. Discrimination of these lines is achieved by use of a direction attribute rather than an orientation attribute. Due to line extraction problems, a single line segment may be broken into several segments. A proposed matching similarity function enables us to perform multiple collinear-line segment matching, instead of merely one-to-one matching.
Experiments using noisy and complicated real image sequences taken with a hand-held camcorder confirm the robustness of our method in difficult tasks.

1. INTRODUCTION

Automatic line segment tracking through an image sequence is a difficult problem in motion estimation for two reasons. First, due to the difficulty in edge extraction, the extracted end points of each line segment are not reliable. Furthermore, a single line segment may 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 line tracking typically consists of two steps: a prediction step and a matching step. There are two popular prediction techniques: Kalman filter-based prediction [4], and methods that apply epipolar constraints for prediction [1, 11]. Kalman filter-based techniques have two main problems: First, they take several frames to obtain reliable results. Second, they have difficulty in setting up the uncertainties for the segment tracking, which is usually tuned by hand. For these reasons they can track only simple objects or scenes on a long image sequence. The problem of using epipolar constraint is that it requires camera calibration or 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 [7, 3]. They include the end points and the orientation of each line, as well as the distance and the overlapping length between line segments. However, these methods are easily confused when the input consists of closely-spaced line segments, which is common in a scene with man-made objects. They are also unable to deal with broken line segments.

We propose a new line tracking method that predicts the motion of line segments reliably based on our hierarchical optical-flow estimation, discriminates closely-spaced line segments accurately by comparing the line directions, and matches multiple collinear line segments by using our similarity function.

There have been many approaches to optical flow estimation reviewed in [2]. However, there remains one big problem: these techniques cannot handle insufficiently textured areas. We solve the problem by using a simple filling operation. Even when line motion prediction is reliable, it is hard to distinguish closely-spaced lines. We introduce a line direction attribute obtained at the edge extraction stage. While many matching functions have already been introduced [3, 4, 7, 11], very few of them deal with multiple line segments that are broken over an image sequence due to the problem of edge extraction. We propose a simple and expandable similarity function to track collinear multiple line segments.

Our method is robust enough to handle real world image sequences taken with 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 an indoor view of a room.

1.1 Overview

Our paper is structured as follows. After reviewing the Lucas-Kanade method for optical flow estimation, we extend it to predict the line segment motion between frames in section 2. Section 3 presents the matching algorithm we use to track broken and closely-space lines. We then present experimental results on real image sequences in section 4. We close with a discussion and a topic of future research.

1.2 Line Extraction

Line segments are extracted from grey-level 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 using Lowe's method [8].

2. OPTICAL FLOW BASED PREDICTION

As a first step, we predict the positions of each line segment in a consecutive frame. We propose an optical flow-based prediction. In order to handle large motions, we use a coarse-to-fine multi-resolution technique. However errors increase in low textured regions. We solve the problem with a dilation-based filling method.

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 availability of tracking confidence [9, 2, 12]. In general, a 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:

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

The Lucas-Kanade method assumes that neighboring pixels in a small window have the same flow vectors. Then we can choose the displacement vector $\mathbf{d}$ with the following equation.

$$\mathbf{d} = \frac{\sum_w g(\mathbf{u})[J(\mathbf{u}) - I(\mathbf{u})]}{\sum_w g(\mathbf{u})^2} \tag{2}$$

where $I(\mathbf{u}) = I(x, y, t)$, $J(\mathbf{u}) = I(x, y, t + \tau)$ and $g(\mathbf{u})$ is the derivative of $I(\mathbf{u})$.

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 can be applied to overcome this problem. 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

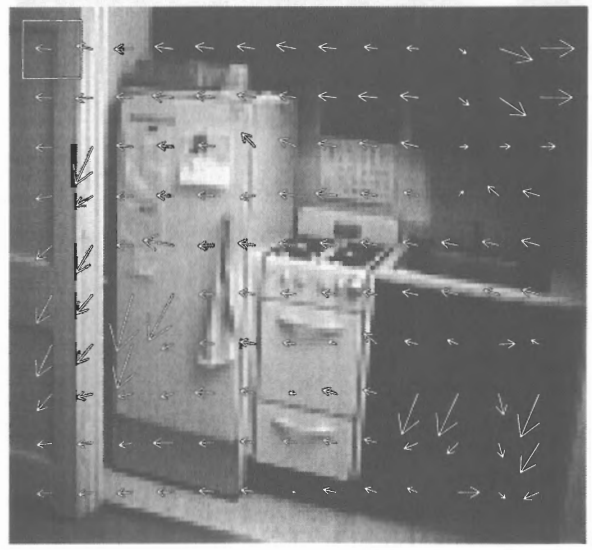


Figure 1: Optical flow with low-textured regions

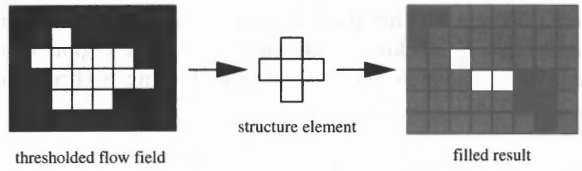


Figure 2: Dilation

surrounded by low-textured regions tend to be smoothed over. Figure 1 shows a result containing insufficient texture.

Poelman solved this problem to some extent in [10]. He calculated the tracking confidence for each region using its trackability and its intensity residue. The trackability can be obtained from its image derivatives [12]. He weighted the results between two levels of the pyramid, according to the confidences. However, the low texture problem still remains.

2.3 Filling by Dilation

We propose a filling method after thresholding at an intra-level of a pyramid, rather than smoothing or weighting. Our method uses 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, our method thresholds the fields with their confidences, and discards poor estimates. We assume that regions containing insufficient texture have similar flow results to their neighboring regions that have reliable estimates. Analogous to a dilation technique in morphology for filling holes in binary images, unreliable optical flow fields can also be filled by using the dilation technique.

Figure 2 illustrates the operation of flow vector dilation. Darker regions represent more reliable flow results and white regions are below the thresholded confidence

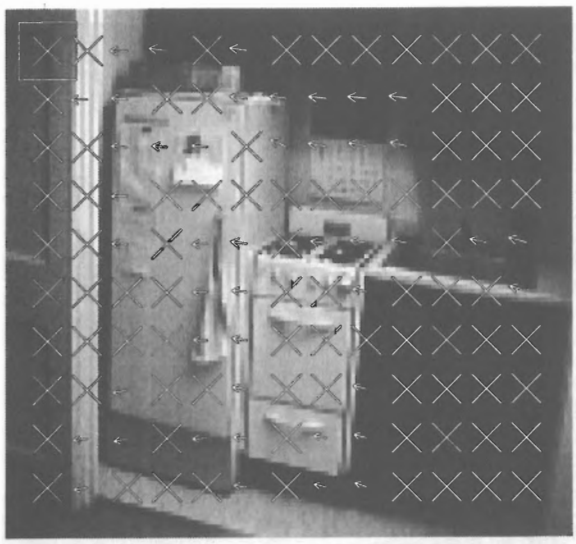


Figure 3: Thresholded optical flow

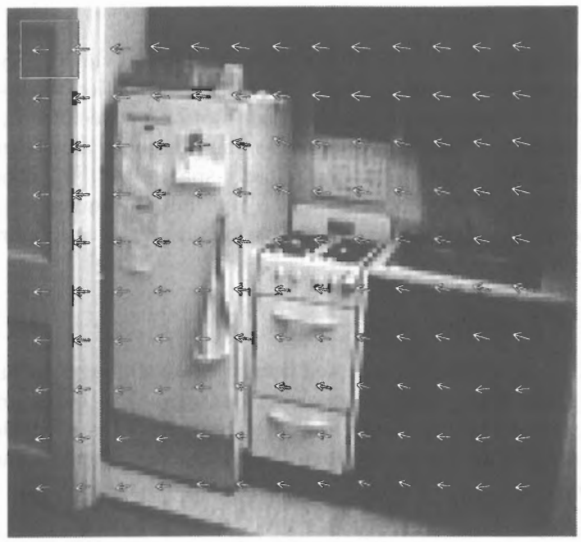


Figure 4: Optical flow using dilation

level. We can fill the blank regions by using the dilation operation. A tracking confidence, $\tau(i,j)$ at region i, j, with its confidence $\gamma(i,j)$ is obtained using a threshold T.

$$\tau(i,j) = \begin{cases} \gamma & \text{if } \gamma > T \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In a region where its tracking confidence is below the threshold, its flow vector is recalculated with its neighboring results. For example, a flow vector, $u(i,j)$, and its confidence, $\tau(i,j)$, at region i, j, are obtained from its four neighboring results weighted by their confidence τ.

$$u(i,j) = \sum_{p,q \in w} \frac{\tau_{pq} u(i+p, j+q)}{\tau_A} \tag{4}$$

$$\tau(i,j) = \sum_{p,q \in w} \frac{\tau_{pq}}{N} \tag{5}$$

where N is the number of regions whose confidences are above the threshold T in its four neighboring regions and

$$(p,q) = (-1,0),(0,-1),(0,1),(1,0), \tag{6}$$

$$\tau_{pq} = \tau(i+p, j+q), \tag{7}$$

$$\tau_A = \sum_{p,q \in w} \tau(i+p, j+q). \tag{8}$$

The operation continues until all fields are filled with the properly estimated values.

Figure 3 shows an example of thresholded flow results. X marks indicate that the regions are thresholded out because they have poor results. Figure 4 shows the filled optical flow field. In regions containing little or no image texture, the flow results have been filled with the results of surrounding highly textured regions.

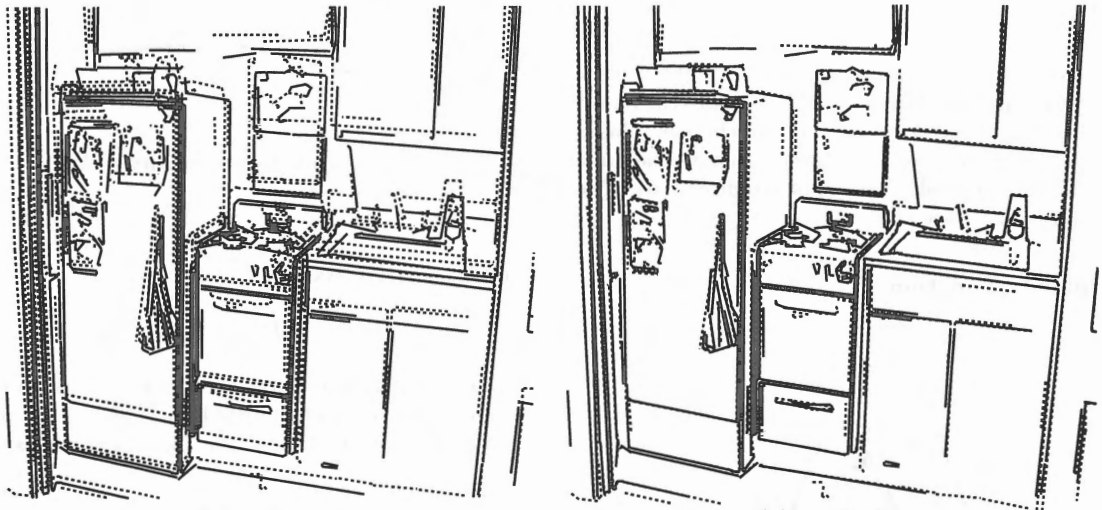
By using this filling-based hierarchical optical-flow estimation at all levels of the pyramid, we can obtain a good prediction for automatic line-segment matching between a pair of images. One advantage of our method is that we do not have to do anything other than set the threshold value. This value depends on images, but it is not difficult to find an appropriate value. In our experiments, we used 100 as the value T in the equation (3).

2.4 Line Segment Motion Prediction

By using the optical flow estimation technique described above, we can predict line segment motion over images. Figure 5 (a) shows line segment motion (i.e., end point position difference) caused by camera motion from one frame to the next. The solid lines represent line segments in the current frame, and the dotted lines represent line segments in the previous frame. We can recognize that there is some amount of motion between the frames. Figure 5 (b) shows the predicted line segment positions with our improved optical flow estimation technique. The dotted lines represent predicted line segments from the previous frame by our optical flow estimation, and the solid lines represent observed line segments in the current frame.

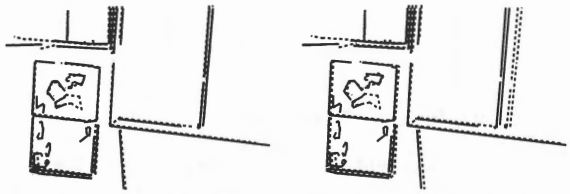
Most of the line segments are correctly predicted with one or two pixel distance errors, although their lengths, determined by the end point positions along the lines, have larger errors due to the line extraction problem. The efficiency of our optical flow estimation can be recognized even in low textured regions, such as the right upper area around the cabinet.

Figure 6 shows a comparison with Poelman's method. Our method was able to predict accurately even in low-textured regions.



(a) difference (b) prediction

Figure 5: Prediction of line segment motion



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

Figure 6: Comparison with Poelman's prediction

3. MATCHING FUNCTION

The second step involves matching the predicted lines from the previous frame to observed lines in the current frame. Man-made objects have a lot of straight line segments. They are often closely-spaced and parallel. Our line direction attribute discriminates these closely-spaced lines. Due to the deficiencies in extracting lines, the end points are not reliable and furthermore the topological connections between line segments are lost during segmentation. Our similarity function solves this problem.

3.1 Line Direction Attribute

Various kinds of line segment attributes have already been introduced for tracking or recognition with line features. For example, Crowley et al. used the following attributes [3] .

θ orientation

h half length

x_c, y_c image coordinates of the center point

a, b, c line equation coefficients

However, these geometric attributes are insufficient for discriminating parallel and closely-spaced line segments.

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 [11]. Still, it cannot distinguish closely-spaced line segments. Because it has difficulty in setting the correct size of a rectangular area to cover the neighborhoods of only a single line. Kanade[6] showed that an edge profile, a curve showing how the intensity changes across an edge, is a good way to find a similar line segment. Setting up a suitable length for extracting an edge profile, however, is difficult when lines are very closely spaced.

We propose another attribute, the line direction attribute, rather than the orientation, which we can obtain from the edge images. This denotes which side of a line is brighter. The edge extraction algorithm gives us not only 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 difficulty in setting up 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 that represents not only the orientation, but also which side is brighter in the intensity image. For example, the direction of a line segment $\mathbf{r}$ can be obtained by averaging the directions $\mathbf{r_i}$ of all edgels belonging to the line as follows:

$$\mathbf{r} = \frac{1}{n} \sum_{i}^{n} \mathbf{r_i} \tag{9}$$

We can obtain the line direction similarity $R(-1.0 \leq R \leq 1.0)$ between two line segments p and q by computing the inner product of their unit direction vectors as follows:

$$R = \mathbf{r_p}^T \mathbf{r_q} \tag{10}$$

We found that this line direction attribute works very well for distinguishing closely-spaced lines in real image sequences as long as the direction does not change. This is because while the most closely-spaced lines have the same orientations, they typically have opposite directions. The computation is also very fast.

3.2 Similarity Function

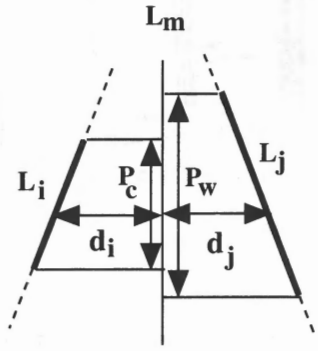


Figure 7: Similarity function

In order to measure the similarity between a predicted line segment and an observed segment, several definitions have already introduced. For example, Crowley et al. proposed the following definition using the line attributes described in 3.1 and their variances in [3].

$$\begin{aligned} sim(L_p, L_o) &= \frac{(\theta_o - \theta_p)^2}{\sigma^2_{\theta o}} \\ &+ \frac{(x_o - x_p)^2 + (y_o - y_p)^2}{(h_p + h_o)^2} \\ &+ \frac{(a_p x_o + b_p y_o + c_p)^2}{\sigma^2_{cp}} \\ &+ \frac{(a_o x_p + b_o y_p + c_o)^2}{\sigma^2_{co}} \end{aligned} \tag{11}$$

where the subscript p represents predictions and o represents observations. The first problem of this definition is that the third and fourth terms depend on the image plane positions. The next problem is that it does not consider broken line segments.

We propose a similarity function which can be applied to matching multiple line segments. Our purpose is to measure the similarity between a predicted line from the previous frame L_i, and an observing line in the current frame L_j. Let us define the distance as D, and the overlapping ratio as P, between the two lines. In figure 7, the middle separating line L_m is the axis of symmetry between the two lines. Let d_i denote the average distance of end points between line L_i and the middle line L_m. Let P_c denote the length of overlapping part of the projected lines to the middle line L_m, and P_w denote the length of the whole part. The distance D between the two lines L_i and L_j is as follows:

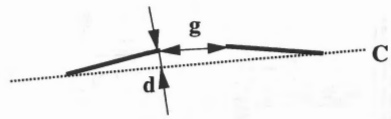


Figure 8: Collinearity test

$$D = d_i + d_j \tag{12}$$

The overlapping ratio P, the ratio between the overlapping part P_c and the whole part P_w of the projected two lines onto the middle line L_m, is defined as follows:

$$P = \frac{P_w}{P_c}. \tag{13}$$

Intuitively, when two lines are similar, the distance D is small and the overlapping ratio P is close to unity. Thus we will define similarity $s(i,j)$ as follows:

$$s(i,j) = \frac{1}{k_d D + k_p P} \tag{14}$$

where k_d and k_p are coefficient parameters set by the user.

3.3 Multiple Line Matching

A single line segment may be broken into multiple segments during the edge extraction stage. Previous methods have not been able to solve this problem. Here we show how our similarity function can be extended to the case of matching multiple line segments.

Figure 9 (a) illustrates an example. When line L_k is collinear to line L_j and the gap is reasonably small, we can consider them to be a single-combined line segment. These line segments are checked and registered beforehand as possible combinations to form a line segment.

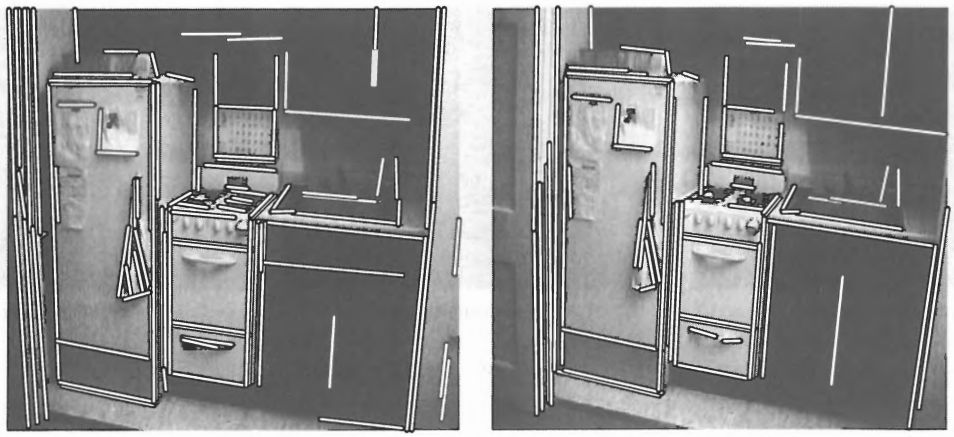
In order to examine the collinearity of line segments, we use the following test procedure. Figure 8 shows the distance d between the collinear line C and the end points, and the gap g between two line segments. We calculate the ratio G_r between the gaps g and the sum of line lengths L, and the maximum distance C_d from the collinear line C as follows:

$$G_r = \sum_i \frac{g_{i-1}}{L_i}, \tag{15}$$

$$C_d = max(d_i). \tag{16}$$

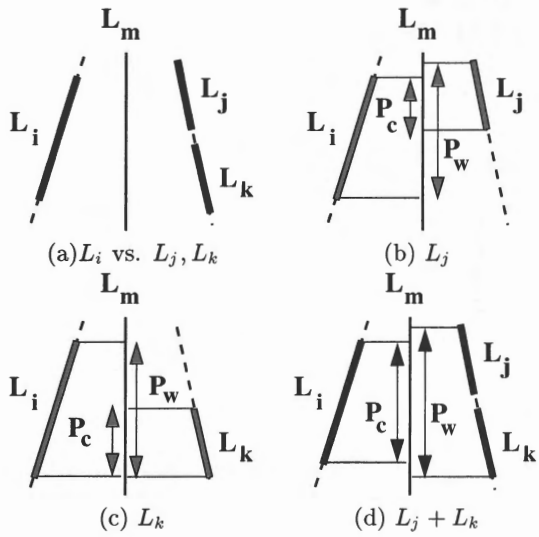
We consider line segments as a potential single-combined line, when their gap ratio G_r and the maximum distance C_d are smaller than pre-defined amounts. We use $G_r = 0.1$ and $C_d = 2.0$ as thresholding values in the following experiments.

Figure 9 (b), (c) and (d) illustrates the possible matching patterns. In order to find the most similar line for L_i, we examine not only the single lines L_j and L_k separately, but also the combined line of L_j and L_k.



(a) selected lines (b) the tracked lines

Figure 10: Tracked result (Kitchen)



(a)L_i vs. L_j, L_k (b) L_j

(c) L_k (d) $L_j + L_k$

Figure 9: Multiple segment matching

4. EXPERIMENTS

This section presents the experimental results achieved by applying our line tracker to both indoor and outdoor real image sequences, taken with a hand-held camcorder. They were digitized on an SGI Indy.

4.1 Kitchen Sequence

Figure 10 shows the result of the kitchen sequence with 10 frames. In the first frame, 384 line segments were obtained using the extraction method described in the introduction. Figure 10 (a) shows 100 lines automatically selected according to their edge strengths and line lengths. Figure 10 (b) shows the tracked line segments in the 10th frame. Out of 100 line segments, 93 line segments were correctly tracked. 85 of them were visible over the entire sequence and tracked correctly, 4 of them were properly detected out of the field. The disappearance of 4 of them (because of line extraction) was properly detected. The main reason of the false tracking was imperfection of edge extraction.

4.2 Building Sequence

Figure 11 is another example of the tracking results. This sequence has a greater number of line segments, larger motion, and a greater number of frames (16 frames) than the kitchen sequence. Due to the large range of motion, there is large discrepancy 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 tower on the left 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 a moving camera, this sequence includes moving people and tree shadows.

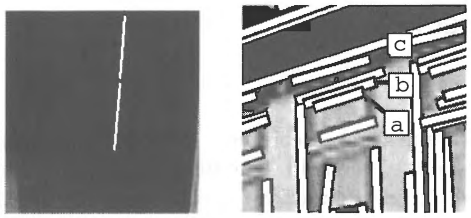
In the first frame, 610 line segments were extracted and 200 of them were automatically selected in the same way as the kitchen sequence. Figure 11 (a) shows the selected line segments. In the second frame, 190 of those line segments were correctly tracked. The tracker could detect all of the existing 12 out of field lines, but only 2 of the 5 lines that disappeared. Most of the wrongly tracked lines were either around the tree shadow at the bottom left or the moving people. It is not necessary to track them for our purpose. In the last frame, 48 line segments remained and all of them were correctly tracked.

4.3 Broken and Closely-Spaced Lines

These experiments included tracking of broken lines and closely-space lines. Figure 12(a) shows an example of correctly tracked broken lines. The line was a single line

(a) first frame (b) second frame (c) 16th frame

Figure 11: Tracked result (Building)



(a) broken line (b) closely-spaced lines

Figure 12: Details of the tracked results

in the previous frame, but it is broken into two lines in the current frame with its 3 pixel gap.

Figure 12(b) shows an example of correctly tracked closely-spaced lines. This is the upper right part of the building sequence around the center window on the second floor of the closer building. The line segments a, b, and c are very closely-spaced and separated by as few as 5 pixels. Although they have similar lengths and orientations, the directions of a and b, and the directions of b and c, are opposite. Therefore all three of them are correctly tracked by our method until they went out of the field of view.

4.4 Comparisons with Previous Work

We evaluated our tracker by comparing it with previous work. First, we compared our flow estimation with Poelman's method. The estimation using the two above mentioned types of data is shown in table 1. It shows the number of correctly tracked lines. For both types of data, our method is better. Next, we compared our matching method with Crowley's method. Our method is better, especially for the kitchen data.

4.5 Determining the Matching Coefficients

The matching coefficients in equation (14), k_d and k_p describe the distance and the overlapping ratio between two line segments respectively. We observed the number of wrongly tracked lines, while we changed the value of k_p and set the k_d as $1 - k_p$ (Figure 13) . The number of

Table 1: Comparison with previous methods

Flow Est.	Matching	kitchen	building
our method	our method	93	190
Poelman	our method	90	185
our method	Crowley	59	173

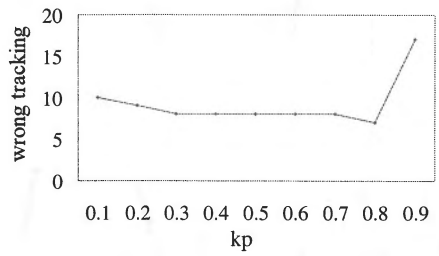


Figure 13: Weighting factors and the wrong tracking

wrong tracking increases dramatically when k_p is above 0.8. We noticed that we should use a value between 0.2 and 0.8. In our experiments, we used $k_d = 0.3$, $k_p = 0.7$ as the coefficients.

4.6 Example of Multiple Segment Tracking

Figure 14 shows a more complicated example of tracking multiple segments. It shows two frames of a 15 frame sequence. The lines on the edge were broken into several segments such as 2, 1, 2, 4 during the sequence. Our method was able to track them correctly.

The experiments show that very good results are obtained using real image sequences. In the kitchen sequence, our method could even track line segments surrounded by low-textured regions and broken line segments due to the edge extraction problem. In the building sequence, we showed that the tracker is robust enough to distinguish closely-spaced lines.

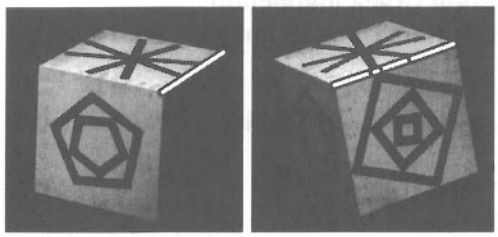


Figure 14: Example of multiple-segment matching

5. DISCUSSION

Our tracker improves robustness in both prediction and matching steps. In the prediction step, we showed that our dilation-based optical flow estimation gives more accurate line position predictions than previous methods. This could be applied to point-feature (small rectangular area) tracking as well.

Kalman filter-based methods are another alternative for prediction. They have the following problems compared with optical flow-based methods. First, they require more than two frames. Second, they need to tune the uncertainty by hand for updating the status. This strongly affects the tracking performance [3]. The accuracy comparison between the two approaches is future work.

In the matching step, we defined a similarity function which is independent of the image plane position. We showed that using edge direction is helpful to discriminate closely-spaced lines. Our similarity function considers multiple-segment matching and works well. We showed that our tracker is more robust with these techniques than previous methods on real image sequences. This tracker is used to shape recovery in [5].

6. CONCLUSION

This paper has addressed the problem of tracking line segments over an image sequence, especially those with unknown motion, taken with a hand-held camcorder. We have demonstrated that our line segment tracker is suitable for noisy and complicated real-image sequences. The tracking method developed in this paper has several advantages over existing techniques. Most notably it has the ability to track broken lines and closely-spaced lines.

The method includes three novel techniques. First, by using our dilation-based hierarchical optical-flow estimation technique, we obtain an accurate prediction of line segment motion between frames, even with large-motion and low-textured regions. Second, we showed that the line direction attribute is helpful in distinguishing closely-spaced lines when we match line segments between frames. Third, the proposed similarity function can overcome imperfections of edge extraction and handles broken line segments in real image sequences.

The results in this paper indicate that using grey-level information from the original images at all stages, even after extracting line segments, improves tracking accuracy. In future work, we plan to incorporate color information.

Acknowledgements

We would like to thank Daniel Morris and Teck Khim Ng who read the manuscript carefully and gave us useful comments. We also would like to thank Dr. Masato Osumi, the general manager of Mechatronics RC at SANYO, Mr. Masaaki Hitomi, department senior manager, and Mr. Hidefumi Matsuura, laboratory manager, who gave us the opportunity to present this research.

7. REFERENCES

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

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

[3] Crowley, J., Sterlmaszyk, P., Skordas, T., and Puget, P., 1992. Measurement and Integration of 3-D Structure by Tracking Edge Lines. International Journal of Computer Vision, 8(1), pp.29-52.

[4] Deriche, R. and Faugeras, O., 1990. Tracking line segments. In: 1st European Conference on Computer Vision, Antibes, France, pp. 259-268.

[5] Morris, D. and Kanade, T., 1998. A Unified Factorization Algorithm for Points, Line Segments and Planes with Uncertainty Models. In: International Conference on Computer Vision, Bombay, India, pp. 696-702.

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

[7] Lowe, D., 1987a. The Viewpoint Consistency Constraint. International Journal of Computer Vision, 1, pp. 55-72.

[8] Lowe, D., 1987b. Three-dimensional object recognition from single two-dimensional images. Artificial Intelligence, 31, pp. 355-395.

[9] Lucas, B. and Kanade, T., 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. In: International Joint Conference on Artificial Intelligence, Vancouver, Canada, pp. 674-679.

[10] Poelman, C. , 1995. The Paraperspective and Projective Factorization Methods for Recovering Shape and Motion. Technical Report CMU-CS-95-173, Carnegie Mellon University, Pittsburgh, PA-USA.

[11] Schmid, C. and Zisserman, A., 1997. Automatic Line Matching across Views. In: Computer Vision and Pattern Recognition, Puerto Rico, USA, pp.666-671.

[12] Tomasi, C. and Kanade, T., 1991. 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, Pittsburgh, PA-USA.