Detecting and tracking linear features efficiently

J. C. Clarke, S. Carlsson† and A. Zisserman
Department of Engineering Science, University of Oxford,
Parks Road, Oxford, OX1 3PJ, UK
†Dept. of Numerical Analysis and Computing Science,
Royal Institute of Technology, S - 100 44 Stockholm, Sweden
email: [johnc, az]@robots.oxford.ac.uk, stefanc@bion.kth.se *

Abstract

An efficient method for detecting and tracking linear features in images is described. This novel algorithm combines a sparse sampling of the image data with a RANSAC grouper, and is particularly suited for frame-rate vision. The detector is tunable for both the scale and orientation of the desired line segments. Experimental results demonstrate fast, robust and accurate feature detection and tracking.

1 Introduction

The detection of lines in an image has a long tradition in the computer vision literature. Methods generally begin with a detection of edgels in the image, and then grouping these edgels into lines. A Hough transform approach is often used (eg. [10]). An alternative is grouping the edgels into chains and segmenting the chains into piecewise linear segments eg. by using a "worm", or by a (recursive) orthogonal regression based fit and test cycle (see [3, 8, 12] for a surveys of previous approaches). Once line segments are detected, there are very efficient tracking techniques available — snakes — for following the feature through an image sequence [2, 11]. The line segment is tracked by a small number of linear searches normal to the line at its predicted position.

This paper describes a new method for line segment detection based on an efficient hypothesise and test algorithm. The method is used to detect lines at frame rate on a standard workstation — i.e. it is fast enough to be applied to a reasonably sized region of an image in the 20ms available to process a single field. The lines are then used as visual primitives for tracking, again at frame rate. The main advance is that scale and orientation selectivity are built into the detection process from the start. A consequence is that the detector is "blind" to short line segments, for example, but this is an advantage because we are only interested in line segments which will be suitable for tracking. It is also important to note that

*Thanks are due to Carnegie Mellon University for the use of images from the VASC database. Funding for this work was provided by the EPSRC and EU ACTS Project VANGUARD.

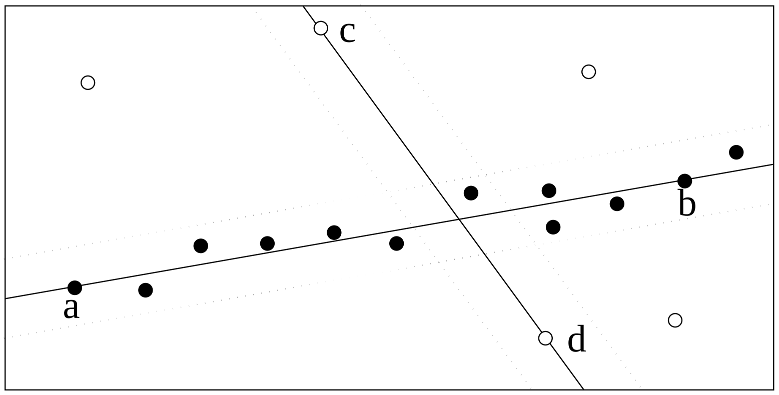


Figure 1: *The RANSAC approach to line fitting in the face of both statistical noise and gross outliers. A number of random samples (here the points $\{a, b\}$ and $\{c, d\}$) are chosen and used to hypothesise lines. Points close enough to the hypothetical lines (within the dotted boundaries) are considered to support the hypothesis. A hypothesis with enough support is accepted – here line $< ab >$ receives 12 votes, while $< cd >$ only has 3. The outliers are denoted by open circles.*

one does not need to detect the entire length of a line segment in order to track it, instead this information is best obtained while tracking.

The next section describes the detection of line segments, while section 3 discusses the tracking of the line segments, and the paper concludes with a critical evaluation of the algorithm.

2 Linear feature detection

The efficient detection of features implicitly demands that one not process every pixel in an image. This section describes a line detector which uses a sparse sampling of the image data to find candidate points (edgels) for lines, and then uses a RANSAC grouper to find line segments consistent with those edgels. RANSAC [9] is a robust fitting algorithm which is finding an increasing number of applications in computer vision, for example in computing the fundamental matrix between two views [7, 14], and for amalgamating line segments [13]. To fit a line passing through a set of points, the RANSAC approach is to randomly select two points (a sample), fit a line through these, and measure the support for the line. The support is the number of points that lie within a threshold distance of the line (the inliers). This process is repeated over a number of samples, and the line with the most support is then chosen. The line fit can then be improved by an orthogonal regression fit to the inliers. If there are several lines present, then there will be several lines with significant support. The algorithm is illustrated schematically in figure 1.

Our algorithm trades increased work in grouping for reduced work in edgel detection by only looking for edgels on a coarse grid of image pixels (every 5th row and column say) and then using RANSAC to find lines through these edgels.

Because edgel detection is the dominant component this tradeoff results in a net gain in speed.

2.1 The algorithm

Finding the edgels

1. A small region (typically 40×40 pixels) of the image is divided into a rectangular grid of widely spaced (5 pixels apart) horizontal and vertical scanlines.

2. Each scanline is convolved with a derivative of Gaussian (here taken as $0.0625 \times [-3, -5, 0, 5, 3]$) to estimate the component of the intensity gradient along the scanline.

3. Any local maxima of the intensity gradient stronger than a threshold (typically $30/256$ pixel value) are considered edgels.

4. The edgels' positions are estimated to sub-pixel precision by taking the maxima of quadratics fitted to each edgel and its two neighbours.

5. The orientation of each edgel is estimated by calculating the intensity gradient normal to the scanline at that point and then taking $\arctan(g_y/g_x)$ where g_x and g_y are the two components of the gradient.

Using RANSAC to find straight lines

1. Two randomly chosen edgels, whose orientations are compatible (within $67.5°$) with the line joining them, are used to hypothesise a line.

2. The number of edgels supporting each putative line are counted. To be considered part of a line an edgel must lie close to it (within 0.1 - 0.25 pixels) and have an orientation compatible (within $67.5°$) with the line.

3. Steps 1 and 2 are repeated (typically 25 times) allowing the dominant line to be found.

4. Lines with enough support (at least 4-6 votes) are deemed present in the image, their supporting edgels removed from the set and steps 1 to 3 repeated until all such lines have been found.

5. The detected lines are re-estimated by an orthogonal regression using all their supporting edgels for increased accuracy.

2.2 Experimental results

This section presents the results from some tests of the new algorithm on images from the VASC database at Carnegie Mellon University and from images acquired in real time by a Sun Ultrasparc computer equipped with an S2200 framegrabber. Figure 2 shows the results of applying the algorithm to grey level images from the VASC database. In each image the detector has been applied sequentially to a series of 40×40 pixel windows tiling the image since this is the size of region

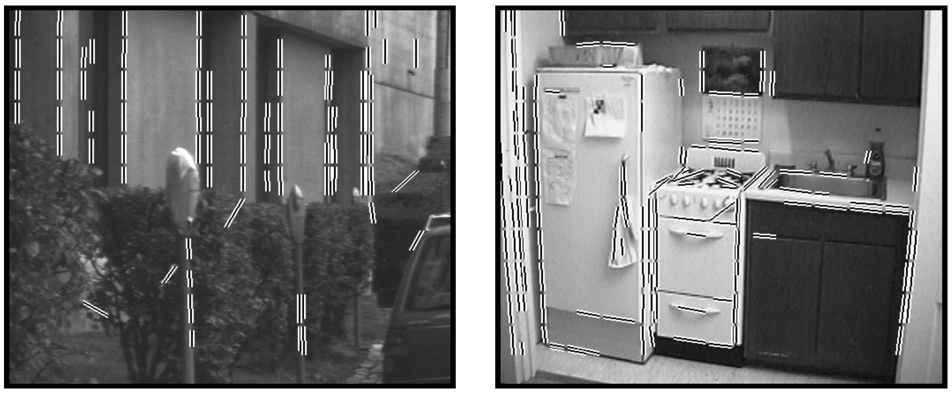


Figure 2: *Images of a parking lot and kitchen scene from the VASC database. The detected line segments have been superimposed on the images. Long lines in the images have been detected as series of short line segments because the line detector has been applied to a series of* 40×40 *pixel windows to duplicate the effect of real time processing. Section 3 explains how the entire extent of the line is recovered during tracking.*

typically processed in real time (20 ms). Notice that the dense texture of the hedge in the parking lot image gives rise to only two false detections. The boundaries of the shadows in the kitchen scene are not detected as lines because they are too blurred and so the maximal gradient edgels do not lie on a line to the required tolerance.

The probabilities of correctly and falsely identifying a line were estimated using 10000 images of two scenes: the first scene consisted of a horizontal black/white boundary, and the second a blank sheet of paper. Each of the 10000 images of each scene was corrupted by the unavoidable pixel noise ($\sigma \approx 3/256$ pixel value) in the camera and framegrabber. In 76 of the 10000 images of a horizontal black/white boundary the detector failed to find the line thus giving the probability of detecting a line $P_D = 0.9924$. This failure rate is similar to that of other feature detectors and is largely due to the random displacement of the edgels. No lines were detected in the 10000 images of a blank sheet of paper, as expected (indeed, the thresholds had to be relaxed to: 3/256 pixel intensity difference, minimum of 4 edgels on a line, edgels within 2.0 pixels of the line, before any false detections occurred).

Given that a line has been detected the uncertainty in its estimated parameters should be quantified. To give an example of the accuracy of the line detector the covariance matrix of the parameters of the horizontal line discussed above was calculated. The line is represented as $\{(x, y) | (-\sin\theta)x + (\cos\theta)y + d = 0\}$ with the average parameters of the 9924 samples $(\theta, d) = (2.34^\circ, 192.43)$. The measured covariance matrix for (θ, d) was:

$$\begin{bmatrix} 8.94e-3 & -5.73e-2 \\ -5.73e-2 & 7.87e-1 \end{bmatrix}$$

The values are all very small - the variance of the orientation of the line is less than one hundredth of a degree.

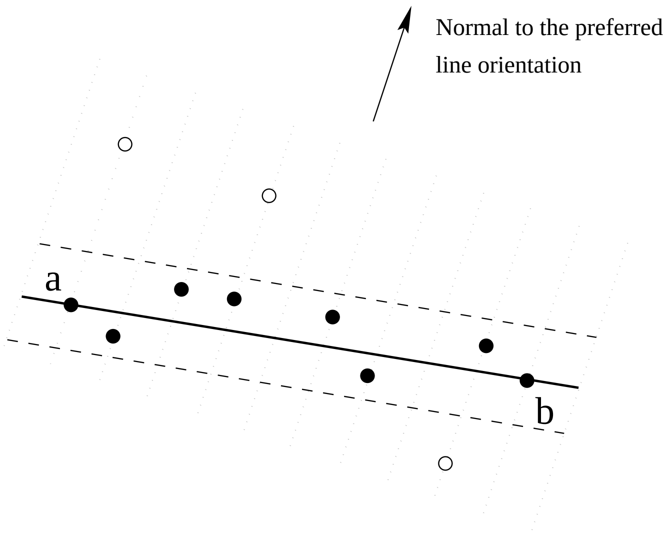


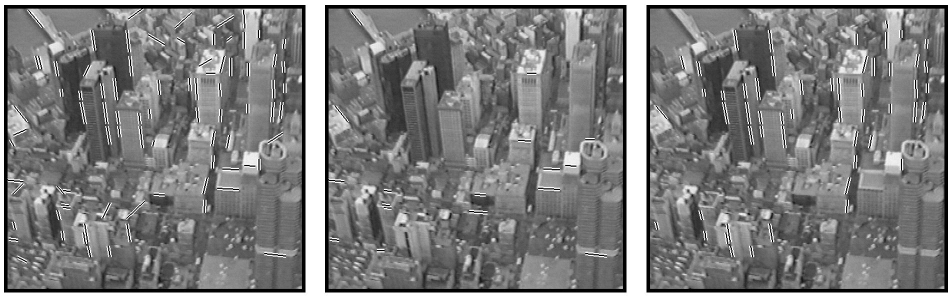
Figure 3: *Efficiently constraining the orientation of detected lines. Edgels (circles) are detected on widely spaced scanlines (dotted lines) all aligned normal to the preferred line orientation. As before two edgels $\{a, b\}$ are used to hypothesise a line whose supporting edgels are counted, and only those hypotheses with enough support are accepted.*

Finally comparing the detector to a standard algorithm performing orthogonal regression fitting lines to edgels detected using Canny's [5] operator showed that the new algorithm runs up to 8 times faster when applied to the same image region and using similar parameters.

2.3 Oriented lines - a steerable detector

Figure 3 shows how, by aligning the scanlines perpendicular to the preferred orientation of detected lines, the method can ignore unwanted lines with no loss of efficiency. The pixels on the (usually diagonal) scanlines can be found easily using Bresenham's line plotting algorithm [4]. The edgels on the scanlines are then processed by the RANSAC grouper as in section 2.1 to extract the line segments. This has much in common with the edge search of a template or snake based tracker although the aim here is to identify line segments when the position and even existence of such lines is unknown *a priori* and must be determined from the image alone, making the use of a robust grouper essential.

The line detector's directional abilities are demonstrated in figure 4. The image is a low resolution aerial view of highrise buildings in Pittsburgh. The ability to detect horizontal and vertical lines is shown (although the detector can be steered to any orientation).



(a) (b) (c)

Figure 4: *An aerial view of Pittsburgh from the VASC database. Images (a), (b) and (c) show the results of isotropic, horizontal and vertical line detection respectively.*

3 Tracking lines

3.1 Predicting a line's position

This section describes how to predict the position of a line in a third view given two previous views and the projection matrices for each view.

Let the image of a world point $\mathbf{X}$ be given by $\mathbf{x} = \mathtt{P}\mathbf{X}$, where, using homogenous coordinates, $\mathbf{x}$ is a 3 vector, $\mathbf{X}$ a 4 vector, and $\mathtt{P}$ the 3×4 projection matrix. Now consider a line $\mathbf{l}$ in the image. All points $\mathbf{x}$ on the line satisfy $\mathbf{l} \cdot \mathbf{x} = 0$, and so

$$0 = \mathbf{l}^\top \mathbf{x} = \mathbf{l}^\top \mathtt{P}\mathbf{X} = \mathbf{\Pi}^\top \mathbf{X} \tag{1}$$

where $\mathbf{\Pi}^\top = \mathbf{l}^\top \mathtt{P}$ is the representation of the plane in the world defined by the optical centre and the image line $\mathbf{l}$. Provided that the planes derived from two views are distinct (i.e. they are not epipolar planes) we are able to find the position of the line in the world by intersecting those planes. If more than two views are available they can all be used in a least squares minimisation to obtain an improved estimate of the line's position.

In practice one deals with line segments while tracking, and so the position of its endpoints in the world must be known. These may be found by backprojecting rays from the endpoints of the line in the image and taking the intersection with the already known position of the line in the world. Since lines do not necessarily intersect in three dimensions the point closest to the backprojected ray may be taken as the endpoint. The position of the line in subsequent images is easily found by projecting its endpoints into the new images.

3.2 Detecting lines

Detection of new line segments: Lines are detected by applying either the isotropic or oriented detector to a randomly chosen small region of the image. If the oriented detector is used the preferred direction for a line is normal to the optical flow in that region.

Predicting lines' positions: Line segments are tracked until they leave the image, or they out-manoeuvre the tracker. Structure based tracking can not be used immediately a line is detected, because at least two views are required. Instead the line segment is tracked using a constant velocity filter until it has moved a small distance in the image. This unguided tracking continues until the structure based tracking can commence.
Locating lines: Once the position of a line has been predicted an attempt can be made to find it. As in snake tracking lines are located quickly by a number (typically 5-7) of short (20 pixels) transverse linear searches. Unlike the snake approach all the strong local maxima of the intensity gradient along the search lines are used in a RANSAC optimisation to find a robust estimate of the true line segment.
Growing line segments: A transverse search is always attempted slightly beyond the currently known extent of the line segment so that the true extent of long lines can be iteratively built up while tracking.

3.3 Experimental results

The line detector/tracker described here has been incorporated into an, originally corner based, tracking system which performs motion segmentation. This system runs on a Sun Ultrasparc equipped with an S2200 framegrabber. Details of the implementation including the feature track life cycle, albeit using corners, can be found in [6].

Figure 5 shows images taken ten seconds apart as the camera translates towards some crayons and blocks. The tracked line segments are shown superimposed in figure 6. Some mismatches and spurious lines are detected but they do not persist because the continued tracking imposes strong constraints on the world structure of features. Figure 7 shows how more than 60 lines are classified as moving in a manner compatible with the camera motion and are tracked using equation 1, while simultaneously up to 20 lines are tracked using the unguided constant velocity filter until they too can be tracked via their 3D structure.

4 Discussion

This paper has described an efficient algorithm for detecting and tracking line segments in images which is able to take advantage of all the constraints on the task. The method has been shown to be fast, accurate and robust to image noise such as dense texture. The main features of the proposed algorithm are:
Advantages:

- It is extremely fast, being able to detect and track on the order of 100 features at frame rate, further the true extent of the line segments is built up continuously.
- The detector is explicitly tunable for both the scale and orientation of the desired line segments.

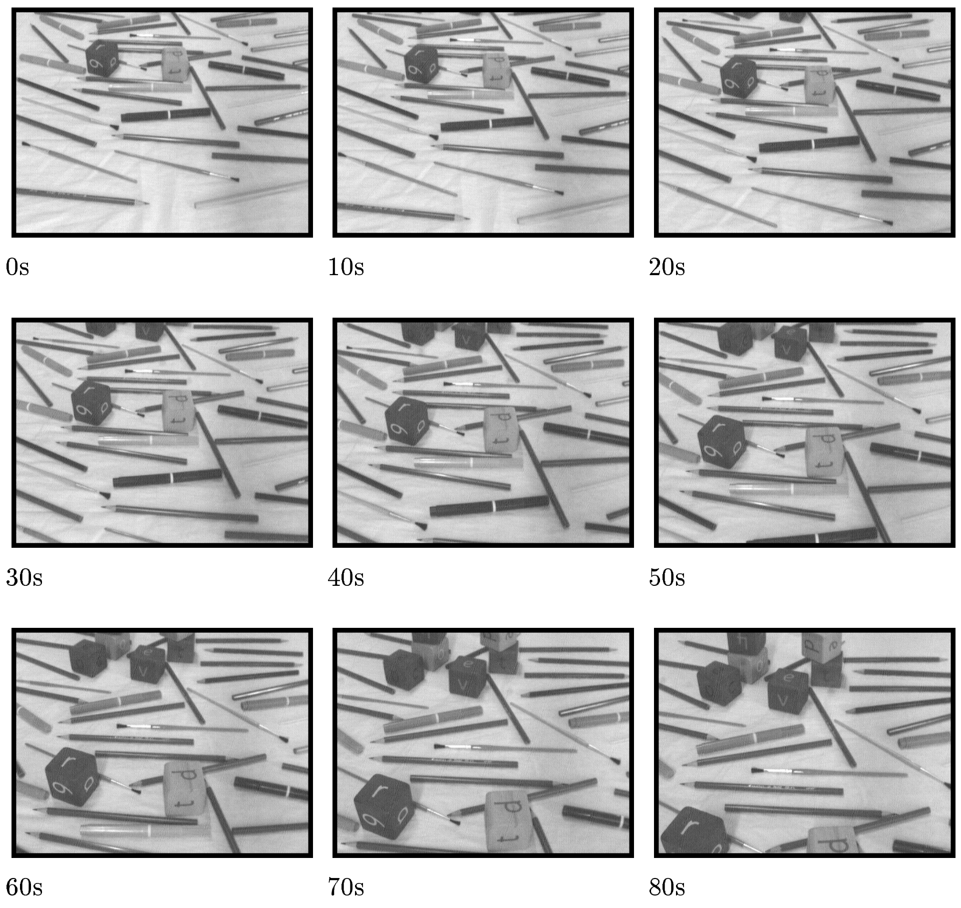


Figure 5: *The computer's view during the first 80 seconds of real time line tracking. A camera mounted on an Adept industrial robot translates forward and down towards some crayons and blocks. (Tracked line segments are superimposed in figure 6.)*

Disadvantages

- Because edgels are detected on a rectangular grid there is an inherent bias against diagonal line segments hence the line detection can not be made perfectly isotropic.

It is important for a feature tracker to be able to choose the orientation of the lines it detects because lines perpendicular to the direction of optical flow will constrain the ego-motion most accurately. Similarly it is useful to ignore very short line segments that are not suitable for tracking.

The RANSAC based detection scheme has here been utilised for line segments. A similar "scale tuned" approach could be used for conic detection. Already RANSAC has been used to improve the efficiency of tracking for both lines and conics when a model exists [1].

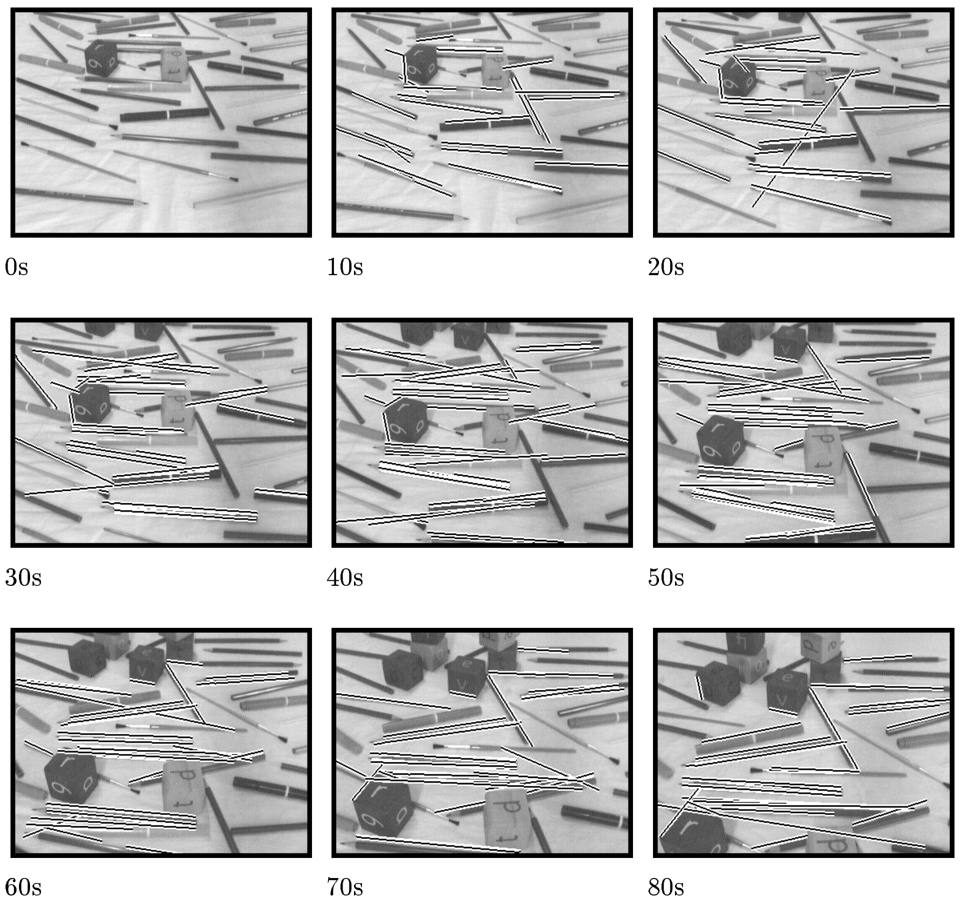


Figure 6: *Results from the first 80 seconds of real time line tracking. (The original images are given in figure 5.) Features whose motion is believed to be consistent with the camera's motion are superimposed on the images. Notice how the number of features tracked increases with time. A few spurious features and matches do occur, but they are quickly removed.*

References

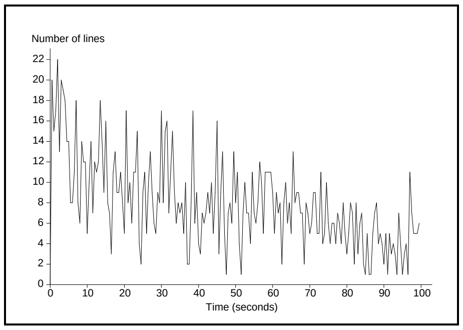
[1] M. Armstrong and A. Zisserman. Robust object tracking. In *Proc. ACCV*, 1995.

[2] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–146, October 1993.

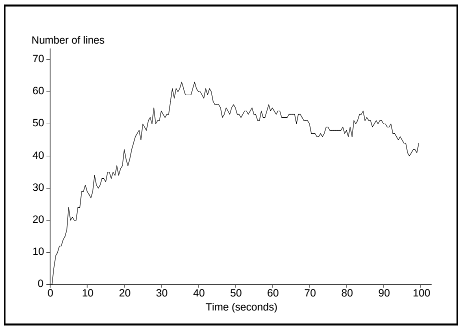
[3] R.N. Bracewell. *Two-dimensional imaging.* Prentice-Hall International, London, 1995.

[4] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 1965.

[5] J.F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.



(a)



(b)

Figure 7: *The number of lines tracked in real time. The number of newly detected line segments undergoing unguided tracking is shown in (a). The number of lines whose motion is consistent with the camera's motion and so are being tracked using their 3D structure is given in (b). As can be seen a total of over 50 lines can be tracked at 50Hz.*

[6] J. C. Clarke and A. Zisserman. Detection and tracking of independent motion. *Image and Vision Computing*, 1996. to appear.

[7] R. Deriche, Z. Zhang, Q.-T. Luong, and O. Faugeras. Robust recovery of the epipolar geometry for an uncalibrated stereo rig. In *Proc. 3rd European Conf. on Computer Vision, Stockholm*, volume 1, pages 567–576, May 1994.

[8] O.D. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, 1993.

[9] M.A. Fischler and R.C. Bolles. Random sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[10] J. Illingworth and J. Kittler. A survey of efficient Hough transform methods. In *Proc. 3rd Alvey Vision Conf., Cambridge*, pages 319–326, 1987.

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proc. 1st Int. Conf. on Computer Vision*, pages 259–268, 1987.

[12] V.S. Nalwa. *A guided tour of computer vision*. Addison-Wesley, Reading, Massachusetts, 1993.

[13] I. D. Reid and A. Zisserman. Goal-directed video metrology. In B. Buxton and R. Cipolla, editors, *Proc. 4th European Conf. on Computer Vision, Cambridge*, volume 2, pages 647–658, April 1996.

[14] P. H. S. Torr and D. W. Murray. Stochastic motion clustering. In *Proc. 3rd European Conf. on Computer Vision, Stockholm*, 1994.