Detecting 3-D Parallel Lines for Perceptual Organization*

Xavier Lebègue and J. K. Aggarwal

Computer and Vision Research Center, Dept. of Electrical and Computer Engr., ENS 520, The University of Texas at Austin, Austin, Texas 78712-1084, U.S.A.

Abstract.

This paper describes a new algorithm to simultaneously detect and classify straight lines according to their orientation in 3-D. The fundamental assumption is that the most "interesting" lines in a 3-D scene have orientations which fall into a few precisely defined categories. The algorithm we propose uses this assumption to extract the projection of straight edges from the image and to determine the most likely corresponding orientation in the 3-D scene. The extracted 2-D line segments are therefore "perceptually" grouped according to their orientation in 3-D. Instead of extracting all the line segments from the image before grouping them by orientation, we use the orientation data at the lowest image processing level, and detect segments separately for each predefined 3-D orientation. A strong emphasis is placed on real-world applications and very fast processing with conventional hardware.

1 Introduction

This paper presents a new algorithm for the detection and organization of line segments in images of complex scenes. The algorithm extracts line segments of particular 3-D orientations from intensity images. The knowledge of the orientation of edges in the 3-D scene allows the detection of important relations between the segments, such as parallelism or perpendicularity.

The role of perceptual organization [5] is to highlight non-accidental relations between features. In this paper, we extend the results of perceptual organization for 2-D scenes to the interpretation of images of 3-D scenes with any perspective distortion. For this, we assume *a priori* knowledge of prominent orientations in the 3-D scene. Unlike other approaches to space inference using vanishing points [1], we use the information about 3-D orientations at the lowest image-processing level for maximum efficiency.

The problem of line detection without first computing a free-form edge map was addressed by Burns et al. [2]. His algorithm first computes the intensity gradient orientation for all pixels in the image. Next, the neighboring pixels with similar gradient orientation are grouped into "line-support regions" by a process involving coarse orientation "buckets." Finally, a line segment is fit to the large line-support regions by a least-squares procedure. An optimized version of this algorithm was presented in [3].

The algorithm described in this paper is designed not only to extract 2-D line segments from an intensity image, but also to indicate what are the most probable orientations for the corresponding 3-D segments in the scene. Section 2 explains the geometry of

* This research was supported in part by the DoD Joint Services Electronics Program through the Air Force Office of Scientific Research (AFSC) Contract F49620-89-C-0044, and in part by the Army Research Office under contract DAAL03-91-G-0050.

projecting segments of known 3-D orientation. Section 3 describes a very fast algorithm to extract the line segments from a single image and to simultaneously estimate their 3-D orientation. Finally, Sect. 4 provides experimental results obtained with images of indoor scenes acquired by a mobile robot.

2 Motivation and Assumptions

We chose to concentrate on objects which have parallel lines with known 3-D orientations in a world coordinate system. For example, in indoor scenes, rooms and hallways usually have a rectangular structure, and there are three prominent orientations for 3-D line segments: one vertical and two horizontal orientations perpendicular to each other. In this paper, any 3-D orientation is permitted, as long as it is given to the algorithm. Therefore, more complex environments, such as polygonal buildings with angles other than 90 degrees, are handled as well if these angles are known. It is important to note that human vision also relies on prominent 3-D orientations. Humans feel strongly disoriented when placed in a tilted environment.

Vertical lines constitute an interesting special case for two reasons: they are especially common in man-made scenes, and their 3-D orientation can easily be known in the 3-D camera coordinate system by measuring the direction of gravity. If a 2-axis inclinometer is mounted on the camera and properly calibrated, a 3-D vertical vector can be expressed in the 3-D coordinate system aligned with the 2-D image coordinate system. Inexpensive commercial inclinometers have a precision better than 0.01 degree. Humans also sense the direction of gravity by organs in their inner ears. In our experiments, we estimate the third angular degree of freedom of the camera relative to the scene from the odometer readings of our mobile robot. Provided that the odometer is constantly corrected by vision [4], the odometer does not drift without bounds.

We can infer the likely 3-D orientation of the line segments from their 2-D projections in the image plane. With a pinhole perspective projection model, lines parallel to each other in the 3-D scene will converge to a vanishing point in the 2-D projection. In particular, if the orientation of the camera relative to the scene is known, a vanishing point can be computed for each given 3-D orientation before the image is processed. All the lines that have a given orientation in 3-D *must* pass through the associated vanishing point when projected. Conversely, if a line *does not* pass through a vanishing point, it cannot have the 3-D orientation associated with that vanishing point. In practice, if a line *does* pass through a vanishing point when projected, it is likely to have the associated 3-D orientation.

To summarize, the line detection algorithm of Sect. 3 knows in each point of the image plane the orientation that a projected line segment would have if it had one of the predefined 3-D orientations. Therefore, the basic idea is to detect the 2-D segments with one of the possible orientations, and mark them with the associated 3-D orientation hypothesis.

3 Detecting Segments and Estimating their 3-D Orientation

3.1 Coordinate Systems and Transformations

The coordinate systems are W (the World coordinate system, with a vertical z-axis), R (the Robot coordinate system, in which we obtain the inclinometer and odometer readings), C (the Camera coordinate system), and P (the coordinate system used for

the perspective projection on the retina). The homogeneous coordinate transformation matrix from W to R is $T_{WR} = T_{roll} T_{pitch} T_{heading} T_{translations}$. T_{roll} and T_{pitch} are known with a good precision through the inclinometer. $T_{heading}$ is estimated by the odometer and $T_{translations}$ is not used here. T_{RC}, the coordinate transformation matrix from R to C, needs to be completely determined through eye/wheel calibration. Finally, T_{CP} is known through camera calibration.

3.2 Overview of the Algorithm

The processing can be outlined as follows:

1. Line support region extraction: compute the angle between the intensity gradient at each pixel and the expected direction of the projection of each 3-D orientation (see Sect. 3.3 for details). Use a loose threshold to allow for noise in the gradient orientation. Reject improper pixels and 3-D orientations.
2. Non-maxima suppression: keep only the local gradient maxima along the estimated perpendicular to the line.
3. Pixel linking: create chains of pixels using a partial neighborhood search in the direction of the estimated vanishing points. This creates noisy linear chains.
4. Line fitting: perform a least-squares fit of line segments to the pixel chains. Recursively break the pixel chains which cannot be closely approximated with a line segment into smaller chains.
5. Global orientation check: compute the match between each line and each 3-D orientation, like in the line support extraction step but with a much tighter threshold.

If the *a priori* heading is very uncertain, the lines will be extracted with loose thresholds, the true heading will be estimated, and the algorithm can then be run again with tight thresholds for the correct categorization.

3.3 Extracting Line Support Regions

For each pixel in the input intensity image and for each category of possible 3-D orientations, we compute the angle between the intensity gradient and the expected direction of the line in 2-D. The expected line is given by the current pixel and the vanishing point associated with the 3-D orientation. It is not necessary to compute the location of the vanishing point (which may lie at infinity).

The homogeneous transformation matrix changing world coordinates into projective coordinates is $T_{WP} = T_{CP} T_{RC} T_{WR}$. Let $[p_x, p_y, p_z, 0]^T_W$ be a non-null vector in the 3-D direction under consideration. If $[su, sv, s, 1]^T_P = T_{WP} [x, y, z, 1]^T_W$ defines the relation between a 2-D point $[u, v]^T$ and its antecedent by the perspective projection, then

$$[s'u', s'v', s', 1]^T_P = T_{WP} \left([x, y, z, 1]^T_W + [p_x, p_y, p_z, 0]^T_W \right)$$

defines another point of the estimated 2-D line. A 2-D vector $\mathbf{d}$ in the image plane pointing to the vanishing point from the current point is then collinear to $[u' - u, v' - v]^T$. Algebraic manipulations lead to $[d_u, d_v]^T = [a_x - a_z u, a_y - a_z v]^T$ where

$$[a_x, a_y, a_z, 0]^T = T_{WP} [p_x, p_y, p_z, 0]^T_W \quad .$$

Note that a_x, a_y, and a_z need to be computed only once for each 3-D orientation.

The current pixel is retained for the 3-D direction under consideration if the angle between $\mathbf{d}$ and the local gradient $\mathbf{g}$ is 90 degrees plus or minus an angular threshold γ. This can be expressed by

$$\frac{\|\mathbf{d} \times \mathbf{g}\|}{\|\mathbf{d}\| \cdot \|\mathbf{g}\|} > \cos\gamma$$

or equivalently:

$$(d_x\, g_y - d_y\, g_x)^2 > (d_x^2 + d_y^2)\,(g_x^2 + g_y^2)\,\Gamma$$

with $\Gamma = (\cos\gamma)^2$ computed once for all. Using this formulation, the entire line support extraction is reduced to 8 additions and 11 multiplications per pixel and per 3-D orientation. If an even greater speedup is desired, $(g_x^2 + g_y^2)$ may be computed first and thresholded. Pixels with a very low gradient magnitude may then be rejected before having to compute $\mathbf{d}$.

4 Results

The algorithm was implemented in C on an IBM RS 6000 Model 530 workstation, and tested on hundreds of indoor images obtained by our mobile robot. The predefined 3-D orientations are the vertical and the two horizontal orientations perpendicular to each other and aligned with the axes of our building. Figures 1 and 2 show the results of line extraction for one image in a sequence. The processing time is only 2.2 seconds for each 512 by 480 image. Preliminary timing results on a HP 730 desktop workstation approach only a second of processing, from the intensity image to the list of categorized segments. The fast speed can be explained partly by the absence of multi-cycle floating-point instructions from the line orientation equations, when properly expressed.

The lines are not broken up easily by a noisy gradient orientation, because the orientation "buckets" are wide and centered on the noiseless gradient orientation for each 3-D orientation category. The output quality does not degrade abruptly with high image noise, provided that the thresholds for local gradient orientations are loosened. The sensitivity to different thresholds is similar to that of the Burns algorithm: a single set of parameters can be used for most images. A few misclassifications occur in some parts of the images, but are marked as ambiguities.

We have compared the real and computed 3-D orientation of 1439 detected segments from eight images in three different environments. The presence of people in some scenes, as well as noise in the radio transmission of images, did not seem to generate many misclassifications. The most frequent ambiguities occurred with horizontal segments parallel to the optical axis: 1.1 % of them were classified as possibly vertical in 3-D.

5 Conclusion

We have presented a new algorithm for detecting line segments in an image of a 3-D scene with known prominent orientations. The output of the algorithm is particularly well suited for further processing using perceptual organization techniques. In particular, angular relationships between segments in the 3-D scene, such as parallelism or perpendicularity, are easily verified. Knowledge of the 3-D orientation of segments is a considerable advantage over the traditional 2-D perceptual organization approach. The orientation thresholds of the 2-D perceptual organization systems cannot handle a significant perspective distortion (such as the third orientation category in Fig. 2). The independence from the perspective distortion brings more formal angular thresholds to

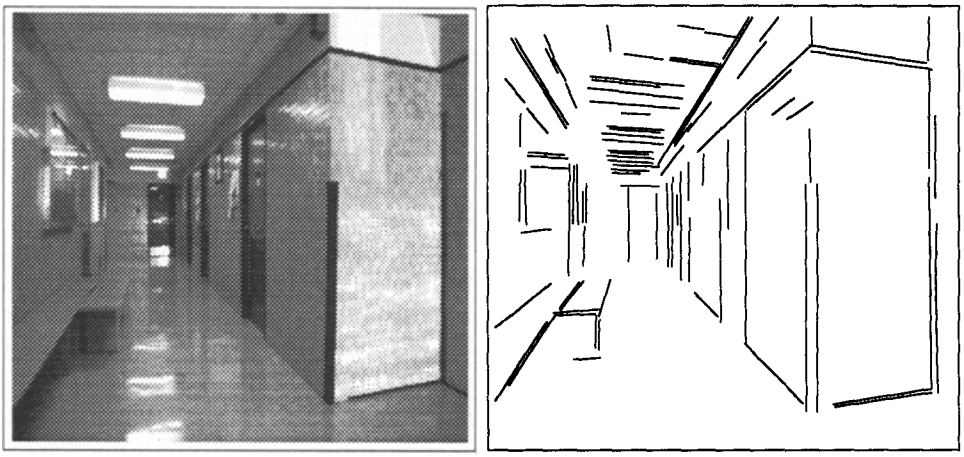

Fig. 1. (a) The input intensity image, and (b) the 2-D segments

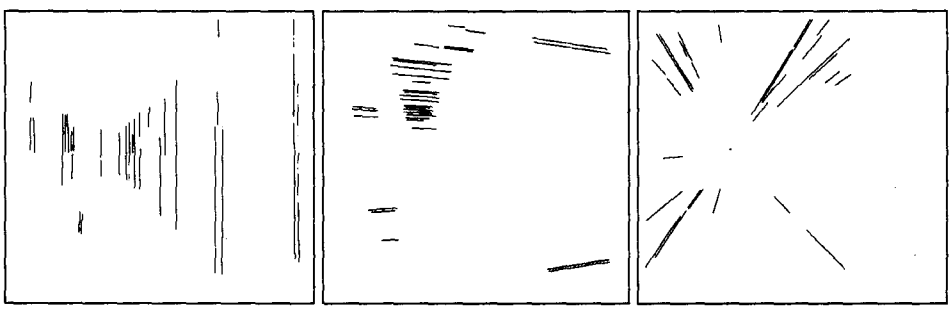

Fig. 2. The line segments associated with each 3-D orientation

the perceptual organization process. By using the 3-D orientation at the lowest image processing level, both the quality and speed of the algorithm were improved. The ultimate benefits of this approach were demonstrated on real images in real situations.

References

1. S. T. Barnard. Interpreting perspective images. *Artificial Intelligence*, 21(4):435–462, November 1983.
2. J. B. Burns, A. R. Hanson, and E. M. Riseman. Extracting straight lines. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(4):425–455, July 1986.
3. P. Kahn, L.Kitchen, and E. M. Riseman. A fast line finder for vision-guided robot navigation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(11):1098–1102, November 1990.
4. X. Lebègue and J. K. Aggarwal. Extraction and interpretation of semantically significant line segments for a mobile robot. To appear in *Proc. IEEE Int. Conf. Robotics and Automation*, Nice, France, May 1992.
5. D. G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, 1985.

This article was processed using the LaTeX macro package with ECCV92 style