EXPLOITING VERTICAL LINES IN VISION-BASED NAVIGATION FOR MOBILE ROBOT PLATFORMS

Jin Zhou and Baoxin Li
Department of Computer Science & Engineering
Arizona State University, Tempe, AZ 85287-8809, USA

ABSTRACT

Vertical line-like structures are abundant in man-made environments, such as the vertical edges of buildings in an outdoor environment and door frames and vertical edges of furniture in an indoors environment. These ubiquitous line or line-like structures may provide useful information for mobile robot navigation. In this paper, we propose an approach to estimating the camera orientation based on detected vertical lines. Further we use the estimated camera orientation in one key problem of mobile robot navigation: ground/obstacle detection. Our method is theoretically well-grounded and yet simple and practical in implementation.

Index Terms— Robot vision systems, homography transform

1. INTRODUCTION

For vision-based mobile robot navigation, two essential tasks are obstacle detection and localization. In many cases, obstacle detection is also directly related to the problem of ground plane detection. Various vision-based approaches have been proposed for the problem of obstacle/ground plane detection (e.g., [1-6]), using various visual cues such as motion, disparity, and color, etc. In man-made environments, there exist a lot of vertical line-like geometrical entities that arise from buildings, boxes, bookshelves, cubicle walls, door frames, etc. These vertical line-like entities (which will be simply called "vertical lines" thereafter) should provide very useful cues in guiding robot navigation. In fact, there have been some attempts on using this type of information. However, in prior work such as [9, 10], the vertical lines were simply used as an ordinary feature, and usually the image plane is assumed to be vertical.

In this paper, we employ vertical lines for vision-based robot navigation. Specifically, we design an algorithm that uses vertical lines for estimating the camera orientation, which can be used to eliminate the assumption of a vertical image plane mentioned above. Furthermore, with the estimated camera orientation, the ground-plane coordinates in the image domain can be directly transformed to the real world ground plane coordinates, which is useful for localization and path planning in robotics.

As a direct application of the proposed algorithm, we apply it to the ground plane detection task, expanding our prior work in [8], where a normalized-homography-based approach was proposed for robust ground plane detection. In this work, we show that the proposed algorithm enables us to by-pass a manual calibration stage that is required in [8].

2. MODELING THE IMAGING SYSTEM

We assume that the mobile robot is navigating on a plane. Without loss of generality, we use the model of Fig. 1 for the imaging system. We define the world coordinates system such that the *y* axis is perpendicular to the ground plane and the origin is at the same height as the robot's camera center. The square box represents the robot. Initially, it is at position C_1, with a rotation matrix R_1. After moving to a new location C_2, the rotation matrix becomes R_2. (R_i is defined in the camera's coordinates system by convention).

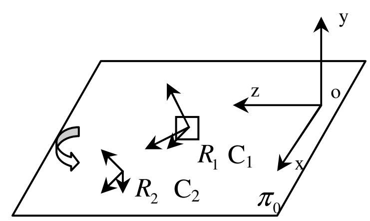

Fig 1. Illustration of the coordinate systems in relation to the ground plane.

Based on the above world coordinates system, we have $C_i = (x_i, 0, z_i)^T$ and the ground plane has coordinates $\pi_0 = (n_0^T, d)^T$ and thus for points on the plane we have $n_0^T X + d = 0$, where $n_0 = (0,1,0)^T$.

Suppose that the internal matrix of the camera is K and that it does not change while navigating, we have the camera matrices as:

$$P_1 = KR_1[I \mid -C_1], \quad P_2 = KR_2[I \mid -C_2] \tag{1}$$

where P_i is a 3×4 matrix, which maps a point in the 3D space with homogeneous coordinates $X = (x, y, z, 1)^T$ to an image point with homogeneous coordinates $\mathrm{x} = (x', y', 1)^T$

$$\mathrm{x} \simeq P_i X \tag{2}$$

If we know K and R_i, we can compute the normalized image coordinates as

$$\tilde{\mathrm{x}} = (KR_i)^{-1}\mathrm{x} \simeq [I \mid -C_i]X \tag{3}$$

If we assume C_i = 0, i.e. assign the origin to be the camera center, then

$$\tilde{x} \simeq (x, y, z)^T \tag{4}$$

For the ground plane points, y is constant, i.e. $-d$. Therefore, we can easily map them between the image coordinates and the real 3D coordinates with (4). However, we need to first know K and R. K may be easily calibrated and can be assumed to be fixed, while R depends on the real pose of the camera, which is consistently changing for a moving robot. This will be our primary concern in subsequent discussion.

3. CAMERA ORIENTATION ESTIMATION FROM VERTICAL LINES

The camera orientation is basically defined by R, i.e. the camera rotation matrix in the world coordinates system. We now show how to use vertical lines to estimate R. Ideally, if the camera's skew is zero and $R = I$, all vertical lines in the physical world will be also vertical in the image. This is intuitive and can be proved as follows.

Suppose a vertical line in the 3D space is represented as $L = (x_0, t, z_0, 1)^T, t \in (-\infty, +\infty)$. Using (4), we can project it to the image domain as

$$l \simeq KR[I \mid -C]L = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0' \\ t' \\ z_0' \end{bmatrix} = \begin{bmatrix} x_0'' \\ t'' \\ z_0'' \end{bmatrix}$$

where x_0'' and z_0'' are fixed and $t'' \in (-\infty, +\infty)$ unless $f_y = 0$.

The above observations lead to the following heuristics: find an R such that vertical lines in the 3D space become also vertical in the image upon rotating back with R. Mathematically, if a camera only rotates, there exists a homography between the old and the new images. In (1), let $C_1 = C_2$ (i.e., rotation only), then we can have

$$I_2 = HI_1, \text{ where } H = KR_2(KR_1)^{-1} \tag{5}$$

where I_i stands for homogeneous coordinates of the image pixels.

In our problem, we can set $R_2 = I$. That is, the camera is rotated back to the world coordinates system. Thus we have

$$H = KR_1^{-1}K^{-1} \tag{6}$$

Since we assume a known K (as in [8]), we can further simplify H by first normalizing the image coordinates as

$$\hat{I}_2 = \hat{H}\hat{I}_1, \text{ where } \hat{I}_i = K^{-1}I_i \tag{7}$$

After the normalization, the orientation of a line will not change since only the origin is translated and the coordinates are scaled. Thus the earlier heuristics still apply, and now we only need to find the normalized homography

$$\hat{H} = R_1^{-1} \tag{8}$$

so that after transformation, all vertical lines in the 3D space become vertical in the image. To find such transformation, we need to solve two problems: (i) to detect lines in the image that are vertical in the 3D space; (ii) find a transform so that all those lines become vertical in the image. We now present our methods for solving these problems.

3.1 Find Vertical Lines

The first problem is to find what lines in the image are vertical lines in the 3D space. While this is in general an ill-posed problem, we simplify the task by making reasonable assumptions based on our application. Suppose that the angle of the vertical lines in the 2D image is zero, then those lines whose absolute angle is less than a threshold is the projection of 3D lines that are nearly vertical. If the rotation around z and x axes is not much, which is usually the case for a fixed camera on a mobile platform, we can use Hough transformation to search for all lines and then filter out vertical lines using the above criteria.

3.2 Find R by Transforming Vanishing Point

Theoretically, all parallel lines in the 3D space intersect at same vanishing point. After projection to a 2D image, the projected vanishing point maybe not at infinity any more, which means the parallel lines are no longer parallel in the 2D image. To make them parallel in the image, we can simply transform the vanishing point to the infinity. For our problem, we need to transform the vanishing point to $(0,1,0)^T$, which is the coordinates of the vanishing point of all parallel vertical lines in the 2D image. Therefore, we can formalize our approach to finding R as: Suppose that the vanishing point in the normalized 2D image is $v \simeq (x_v, y_v, z_v)^T$, find a rotation matrix $\hat{H} = R_v = R_1^{-1}$ such that $R_v v \simeq (0,1,0)^T$. In our approach, R_v is computed by rotating the camera around two axes independently. First, we rotate the camera around its z axis, which is equal to rotating the image coordinates, such that the vanishing point is transformed to the y axis (that is, x = 0). Second, we rotate the camera around the x axis to transform the vanishing point to the infinity. Finally, we have

$$R_v = R_x R_z, \text{ where}$$

$$R_z = \begin{bmatrix} \cos(\theta_z) & \sin(\theta_z) & 0 \\ -\sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ such that } R_z \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} 0 \\ y_v' \\ z_v' \end{bmatrix} \tag{9}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & \sin(\theta_x) \\ 0 & -\sin(\theta_x) & \cos(\theta_x) \end{bmatrix} \text{ such that } R_x \begin{bmatrix} 0 \\ y_v' \\ z_v' \end{bmatrix} = \begin{bmatrix} 0 \\ y_v'' \\ 0 \end{bmatrix}$$

From (9), we have:

$$\theta_z = -\arctan(x_v / y_v), \quad \theta_x = \arctan(z_v' / y_v') \tag{10}$$

3.3 Iterative R_v Estimation

Before estimating R_v, we need to transform the lines to the normalized image space as (7) defines. Suppose that a line is represented as a *3x1* vector $l = (a,b,c)^T$, after transformation, $l' = (K^{-1})^{-T} l = K^T l$ ([7]). Due to noise and inaccuracies in vertical line detection, we need a robust estimation algorithm for R_v. We first define an error measurement for R_v: Given a set of normalized vertical lines $\{l_i\}$ and an estimate $\tilde{R}_v$, the error of $\tilde{R}_v$ is defined as

$$E(\tilde{R}_v) = \sum_i E(l_i \mid \tilde{R}_v) = \sum_i E(l_i' = \tilde{R}_v l_i) = \sum_i \left|\theta(l_i')\right| \quad (11)$$

where l_i' represents transformed line of l_i under $\tilde{R}_v$, and $\theta(l_i')$ is the angle of l_i' (assuming that the vertical line's angle is zero). Intuitively, the error function is defined so as to make all transformed lines to be vertical.

We have proposed the method for estimating R_v in section 3.2, which starts from the vanishing point. Basically, vanishing point is the intersection of the vertical lines. Ideally, all detected vanishing lines should have a common intersection points. But in practice, there may be multiple intersections. Here we propose the following algorithm for searching for the best R_v in the rotation space:

Input: An image I, camera internal matrix K
Output: camera orientation R_v
Algorithm:
1. Use Hough transformation to detect all lines
2. Choose all lines whose absolute angle is less than T
3. Compute all intersection points from any two lines.
4. Determine R_v from intersection point.
5. Choose a best R_v that has the minimum error.
6. Start with R_v, adjust θ_x and θ_z in a range, and find a better R_v.

4. GROUND PLANE DETECTION WITH ORIENTATION ESTIMATION

In ([8]), we proposed a normalized-homography-based approach to detect ground plane from motion. In general, the homography of the plane can be computed as:

$$H = KR_2(I + \Delta C n^T / d) R_1^{-1} K^{-1} \quad (12)$$

where $\pi = (n^T, d)^T$ is the coordinates of a plane so that for points X on the plane we have $n^T \mathrm{X} + d = 0$. If we assume that the camera only rotates around the y-axis, or $R_2 = R_1 \Delta R_y$. We can use the normalized homography:

$$\hat{H} = (KR_1)^{-1} H (KR_1) = \Delta R(I + \Delta C n^T / d) \quad (13)$$

Since the normal vector of the ground plane is determined, i.e. $n_0 = (0,1,0)^T$, normalized ground plane homography has only 3 degree of freedom(DOF), which is much less than a general homography with 8 DOFs. Moreover, ΔR and ΔC can be directly used for localization.

However, to achieve such DOF reduction, camera orientation (R_1) is required. [8] presents a manual way to calibrate the orientation. With automatic camera orientation estimation, not only can we eliminate initial R_1 calibration, but also we can remove the constraint that the camera can only rotate around the y-axis. We now show how to achieve this.

Given the world coordinates system in Fig. 1. R can be decomposed as:

$$R = R_z R_x R_y \quad (14)$$

This means that, from the world coordinates system to R_1, we first rotate around the y-axis, and then around x and z axes. From (9), since $R_v = R_1^{-1}$, we can recover R_z and R_x. Although R_y is left unspecified, it is reasonable since in the world coordinates system, we only require the y axis is perpendicular to the ground plane, meaning that there is a freedom to rotate around the y axis. Thus we can write R_1 and R_2 as $R_1 = R_{1v}{}^{-1} R_{1y}$ and $R_1 = R_{1v}{}^{-1} R_{2y}$. So (12) becomes

$$H = KR_{2v}{}^{-1} R_{2y} (I + \Delta C n^T / d) R_{1y}{}^{-1} R_{1v} K^{-1} \quad (15)$$

Without losing generality, we can assume $R_{1y} = I$, now we can write the normalized homography as

$$\hat{H} = (KR_{2v}{}^{-1})^{-1} H (KR_{1v}{}^{-1}) = R_{2y}(I + \Delta C n^T / d) \quad (16)$$

Note R_{2y} is a rotation matrix around the y-axis, which has exactly the same form and meaning as ΔR in (13). In (16), R_{1v} and R_{2v} are automatically estimated from images, thus no R calibration is required and the constraint of only rotating around the y-axis is removed.

5. EXPERIMENTS

Experiments have been carried out to assess the proposed algorithm. In the following, sample results are presented. The first example illustrates the process of estimating camera orientation with the detection of vertical lines. Fig. 2 is based an indoor sequence captured by a robot platform. Fig. 2 (a) shows the original image with detected vertical lines in green color. (b) shows the rectified view with rotation around the z axis so that the vanishing points lie on the horizontal center of image. (c) shows the rectified view with rotation around the x axis so that the vanishing points are transformed to infinity, i.e. all lines become parallel. (d) shows the results by applying both rotations so that all vertical lines become vertical in the image.

The second sample illustrates the application of the proposed method to normalized-homography-based ground plane detection. Fig. 3 (a) and (b) show the point correspondences. Fig. 3 (c) shows the rectified first view with orientation calibrated with a check board pattern ([8]). Fig. 3 (d) shows the rectified view with automatically estimated orientation from vertical lines in (e). (f) shows the detected ground plane points in green (red ones are thus obstacle points). (g) shows the difference between the second view and the transformed first view using the computed normalized homography. Note the difference between (c) and (d). This is partially due to the inaccuracy in the detection of vertical lines. For example, as (e) shows, the cabinet's edge is not detected (due to a specific parameter setting in line detection) whereas an edge on the backpack is detected although that may not be an actual vertical line. However, this example serves to illustrate that the proposed method works reasonably well despite the lack of prior calibration and in the presence of inaccuracies of the vertical line detection.

6. CONCLUSIONS

In this paper, we introduced a method to automatically estimate the orientation of the camera on a robot platform, using detected image lines that correspond to vertical line structures in the physical world, which are abundant in most man-made environments. We have showed how this estimated orientation can be used in rectifying a view so that the image plane becomes vertical, which is a common assumption in many vision-guided mobile robot systems. We further applied the technique to normalized-homography-based ground plane detection, which previously would demand a fixed camera and an initial orientation calibration. Both theoretical analysis and experimental evaluation have demonstrated the advantages of the proposed method.

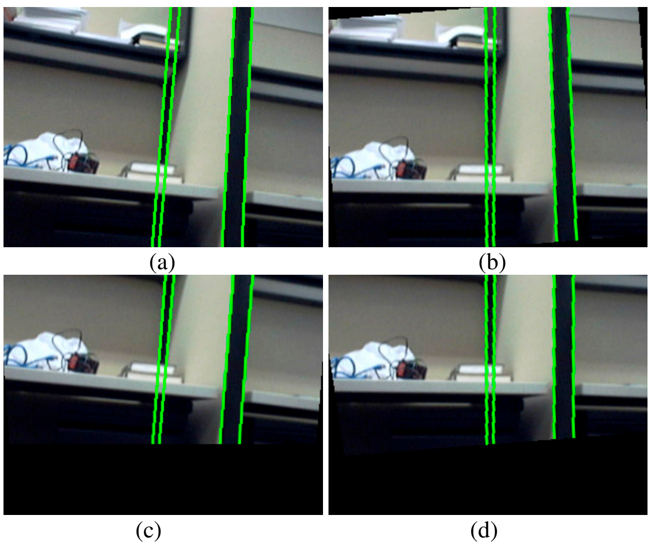

Fig. 2. (a) The original image and detected vertical lines. (b) Rotation around z axis. (c) Rotation around x axis. (d) The results combing (b) and (c), where the detected lines are all vertical and parallel in the image.

REFERENCES

[1] J. Hoffmann, M. Jungel and M. Lotzsch, "Vision Based System for Goal-Directed Obstacle Avoidance",*8th Int. Workshop on RoboCup 2004*.

[2] K. Sabe, M. Fukuchi, J.-S. Gutmann, T. Ohashi, K. Kawamoto, and T. Yoshigahara, "Obstacle Avoidance and Path Planning for Humanoid Robots using Stereo Vision," *Proc. ICRA* 2004, New Orleans, pp. 592 – 597, Vol.1 April 2004.

[3] L. M. Lorigo, R. A. Brooks, and W. E. L.Grimsou, "Visually-guided obstacle avoidance in unstructured environments," *Proc. IEEE/RSJ Int. Conf. Intel. Robot & Systems*, France, vol. 1, pp. 373-379, Sept. 1997.

[4] N. Pears and B. Liang, "Ground plane segmentation for mobile robot visual navigation," *In IROS 2001*, vol. 3, pp. 1513–1518.

[5] B. Liang, N. Pears and Z. Chen, "Affine Height Landscapes for Monocular Mobile Robot Obstacle Avoidance" *Proc. Intelligent Autonomous Systems*, pp. 863-872, August 2004.

[5] S. Lenser and M. Veloso, "Visual sonar fast obstacle avoidance using monocular vision," *In Proc. of the IEEE/RSJ IROS 2003*.

[6] Y. Kim and H. Kim, "Layered ground floor detection for vision-based mobile robot navigation," *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 13-18, 2004.

[7] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2nd edition, 2003.

[8] J. Zhou and B. Li, "Robust Ground Plane Detection with Normalized Homography in Monocular Sequences from a Robot Platform" *In ICIP 2006*.

[9] J.J. Guerrero, C. Sagues, "Navigation from uncalibrated monocular vision", IFAC, *Intelligent Autonomous Vehicles*, pp. 210-215, 1998.

[10] C. Sagues, A. C. Murillo, J. J. Guerrero, T. Goedeme, T. Tuytelaars, L. Van Gool, "Localization with omnidirectional images using the radial trifocal tensor", in *ICRA'06*, pp. 551-556.

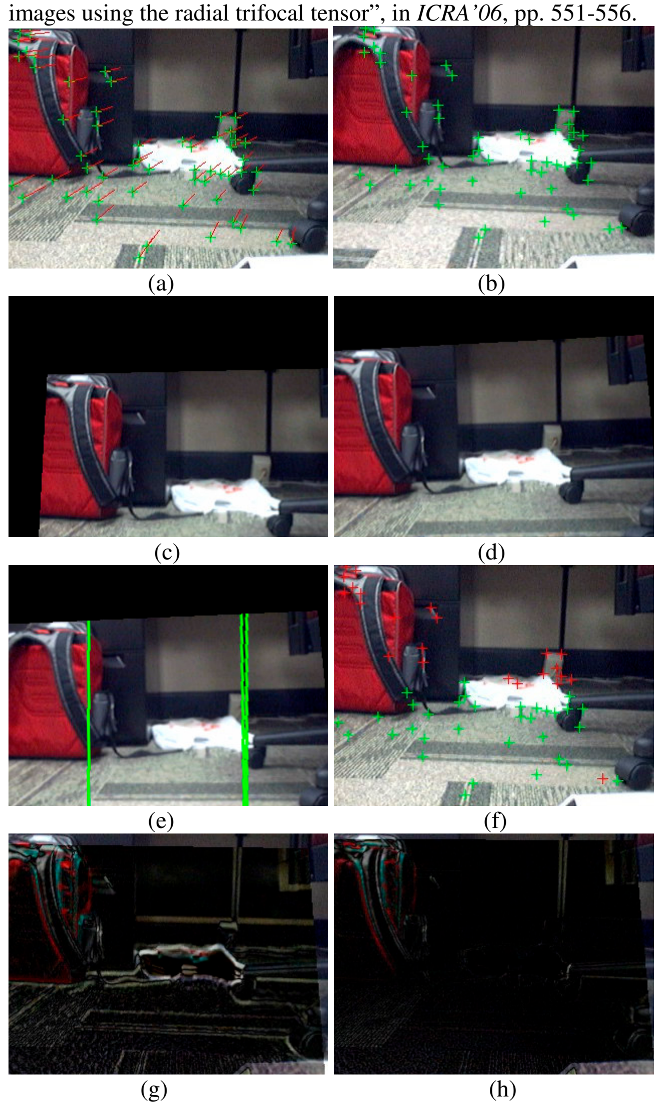

Fig. 3. (a)-(b): two views with feature points and their correspondence (indicated by red line segments). (c): rectified first view with R_1 calibrated by checkboard patterns [8]. (d): rectified view with R_1 automatically estimated from vertical lines in (e). (f): the result of detected ground plane points with R_1 in (d). (g): the difference between the second view and the transformed first view (h): difference using the homography estimated from inliers in (f).