

Fast Vanishing Point Estimation on Smartphones*

Wonwoo Lee[○], Woontack Woo

GIST U-VR Lab.

{wlee, wwoo}@gist.ac.kr

요약

In this paper, we propose a fast vanishing point estimation method for smartphones, which have limited computing power. Our method exploits not only line segments in an image, but also the phone's built-in accelerometer. We first estimate a vanishing point in vertical direction, where the phone accelerometer values are used to initialize the position of the vanishing point. Then, we refine the initial vanishing point by using line segments. The vanishing point corresponding to horizontal lines are estimated from an orthogonality constraint with the previously obtained one. The initial position of a vanishing point provided by the phone accelerometer allows us to reject outliers quickly for vanishing point detection and thus, our method can find vanishing points fast. Moreover, vanishing points are successfully detected in complex environments.

1. Introduction

Vanishing points provide useful information for understanding scenes captured by a camera. They have been used for image rectification, 3D reconstruction, orientation estimation, etc. Gaussian sphere representation [1], [2] have been widely used due to its convenience of handling lines and points. Recent algorithms using line clustering estimate vanishing points in the image space [3], [4]. However, vanishing point detection is still a burden on mobile phones due to their low CPU power.

2. Vanishing Point Estimation

Thanks to the phone accelerometer, we can compute the vanishing points faster on mobile phones by estimating vertical vanishing point from the accelerometer values. We denote by $S = \{s_1, s_2, \dots, s_n\}$ a set of line segments, by v_1 and v_2 the vanishing points corresponding to the horizontal and vertical lines. Line segments of the input image are obtained by using a fast line segment detector [5]. The line segments shorter than 15 pixels are ignored because short lines can cause wrong vanishing point detection.

To compute vanishing points, we define a distance function, which measures the consistency of a line segment s with a given vanishing point v . We adopt the distance function proposed in [6] that defines the distance as the angle between s and the ideal line passing through v and the mid-point of s , as shown in Fig. 1. The distance

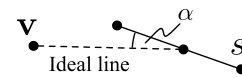


Fig. 1 Distance between a vanishing point v and a line segment s

function is defined as

$$D(s, v) = \cos \alpha \quad (1)$$

which ranges from 0 to 1 and becomes larger if the line of s is more likely to meet v . The vanishing point corresponding to vertical lines can be obtained by projecting the point at infinity $V_2 = (0, 1, 0, 0)^T$ onto the image plane. The vertical vanishing point v_2 is then expressed as

$$v_2 = K \begin{matrix} \hat{e} \\ \hat{e} \end{matrix} R \quad | \quad t \begin{matrix} \hat{0} \\ \hat{0} \end{matrix} V_2 = K V_2^c \quad (2)$$

where the superscript c represents the camera's local coordinate system.

The phone accelerometer provides the direction of gravity g in the phone's local coordinate system. It means g indicates the direction of vertical lines in the camera's local coordinate system, i.e., $g = V_2^c$. Thus, projecting g onto the image plane gives us the vanishing point corresponding to vertical lines directly.

$$v_2 = K V_2^c = K g \quad (3)$$

However, the phone accelerometer values contain noise and we cannot rely on the accelerometer values only. Hence, we compute a rough coordinates of v_2 and build a set of line segment S^v , which is a subset of S . S^v consists of the line segments that are likely to be vertical lines by measuring the distance between a line segment and the initial v_2 . We add a line segment s_i to S^v if the distance

* This research is supported by Ministry of culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA), under the Culture Technology(CT) Research & Development Program 2009.

between s_i and v_2 is below a threshold.

$$S^v = \{s_i^v \mid D(s_i^v, v_2) \leq d_{th}\} \quad (4)$$

Since the initial solution is already close to the exact one, we set $d_{th}=0.99$, which means α in (1) becomes almost 0. After building S^v , we estimate v_2 by using the RANSAC [7] method. The vanishing point that minimizes the mean distance from inliers is chosen as a solution.

For the vanishing point corresponding to horizontal lines, we estimate v_1 through the line clustering method using *J-Linkage* algorithm [3]. Note that only the line segments that are not included in S^v are considered during the process of v_1 estimation.

The first step of *J-Linkage* is building m hypotheses, the candidates for v_1 . The hypotheses are computed as the intersections of two lines randomly selected from available line segments. To keep good candidates and to reject outliers, we apply the orthogonality constraint that two vanishing points should satisfy as defined in (5).

$$K^{-1}v_1 \times K^{-1}v_2 = 0 \quad (5)$$

We keep the intersection of two lines as a candidate if it satisfies

$$K^{-1}v_2 \times K^{-1}p \leq d_{th}, \quad (6)$$

where p is the intersection point of the two lines. δ_{th} controls the tolerance we give to p and $\delta_{th}=0.005$ works well in practice. We compute m hypotheses h_1, h_2, \dots, h_m and a set of lines S^h containing k line segments used for building the hypotheses. Since mobile phones have a low-performance CPU, we set the maximum number of hypotheses to 100 to speed up vanishing point detection. Typically 500 hypotheses are used [3], but 100 hypotheses are enough in our method because we already have a good solution of vertical vanishing point.

The second step is constructing the preference matrix P , a Boolean matrix of $k \times m$. Each row of P represents the preference set of a line segment with respect to the hypotheses. For the i -th line segment, the j -th entry, $P(i, j)$ is computed as

$$P(i, j) = \begin{cases} 1 & \text{if } D(s_i^h, h_j^c) \leq d_{th} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Under assumption that the lines corresponding to the same vanishing point tend to have similar preference sets, we cluster the line segments in S^h . Initially, each lines are considered as a cluster and we measure the similarity between two clusters by the *Jaccard* distance [3]. The preference set of a cluster is the intersection of all preference sets of the lines included in the cluster. All clusters are compared with each other and two clusters having minimum distance are merged. After line segment clustering, we obtain a set of points resulting from the final clusters. Finally, the point minimizing (5) is chosen as v_1 .

3. Experimental Results

We implemented the proposed method on a mobile phone, which has a 800MHz CPU. Typically, 100 to 250

lines are extracted from a real scene and vanishing point detection takes less than 2 seconds. Our method is about 10 times faster than the method in [14], where we set the maximum number of hypotheses to 500 and skip the refinement step because we also do not conduct refinement step. Fig. 2 shows image rectification results based on the detected vanishing points in images. Our method successfully detected vanishing points and the image are correctly rectified. Our method can work with not only complex scenes but also simple scenes, where not many lines exist.



Fig. 2. Image rectification results using vanishing points

참고문헌

- [1] J. Shufelt, "Performance evaluation and analysis of vanishing point detection techniques," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 3, pp. 282–288, 1999.
- [2] A. Almansa, A. Desolneux, and S. Vamech, "Vanishing point detection without any a priori information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 4, pp. 502–507, 2003.
- [3] J. P. Tardif, "Non-iterative approach for fast and accurate vanishing point detection," in *Proceedings of the International Conference on Computer Vision*, pp. 1250–1257, 2009.
- [4] H. Kong, J. Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 96–103, 2009.
- [5] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A Fast Line Segment Detector with a False Detection Control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, 2010.
- [6] C. Rother, "A new approach to vanishing point detection in architectural environments," *Image and Vision Computing*, no. 9-10, pp. 647–655, 2002.
- [7] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.