5E-3

Estimating Geometry from Diffuse Reflection

Wu, Jung-Hsuan † Suguru Saito ††
†Graduate School of Information Science & Engineering, Tokyo Institute of Technology
††Graduate School of Information Sciences, Ochanomizu University
††Graduate School of Information Science & Engineering, Tokyo Institute of Technology

1 Abstract

In this paper, we propose a method to estimate the geometry in an image by the diffuse reflection. Our method requires only a small amount of user inputs and then our system will automatically estimate the geometry of the scene of the input image by minimizing the quadratic pixel-wise difference between the approximated diffuse reflection and the original one.

2 Related Work

Iizuka et al.[3] presented a simple but efficient method to construct the geometry of the scene. Nevertheless, their work assumes that the focal length of the camera is known, which prevents the users who are not familiar with the camera from using their system. The method proposed by Hedau et al.[2] can automatically determine the structure of the room by a single image, however, their work focuses on indoor images and thus is difficult to handle the outdoor scenes. Different from the previous works, our system aims to estimate the geometry under more general cases by requiring the hint which is easier to be specified.

3 Algorithm Overview

A diffuse reflection which is observed on a plane and is caused by a single point light source with Lambertian assumption can be approximated by a normal distribution projected onto the plane. Our system estimates the geometry of the plane by minimizing the difference between the diffuse reflection and the projected normal distribution with requiring only a small amount of user input as described in Sec.3.1. Our system starts with extraction of the diffuse reflection by the method mentioned in Sec.3.2. We propose how to construct the 3D scene by only several parameters in Sec.3.3. Then the parameters which minimize a quadratic energy function are estimated as described in Sec.3.4.

3.1 User Input

Our system requires the user to specify the region which contains the full diffuse reflection by sequentially clicking on outside of the diffuse reflection. Fig.1(left) gives an example of specifying the region by several clicks on the cyan vertex. In addition to the region of the diffuse reflection, our system also allows the user to specify the edges between the ground and the wall. Fig.1(right) shows how the edges are marked by several points.

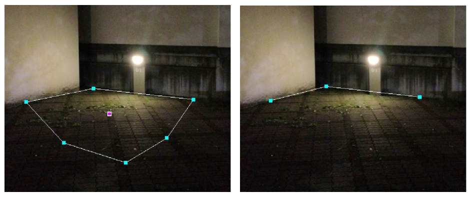
Figure 1: Our user input which specifies the region of diffuse reflection (left), and to mark the edges between the wall and the ground (right).

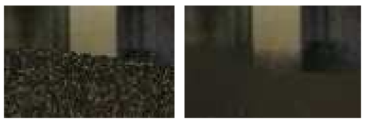
Figure 2: The roughly decomposed reflectance image by inpaiting with nearest neighbor (left) and by inpainting with weighted average (right).

3.2 Rough Intrinsic Images Decomposition

The first step of our system is to roughly estimate M_r, the reflectance component, which is achieved by a modified version of the inpainting technique proposed by Barnes et al.[1]. The user input of diffuse reflection specifies the region to be erased and the pixels which are on the boundary of the region and their adjacent neighbors are the sources to fill the hole. However, we found that filling with the nearest neighbor often generates noisy result (see Fig.2), so we use weighted average of all neighbors to fill the hole, with the weight computed by the inverse of the patch distance used in the previous work [1]. We also add an additional constraint that the neighbors whose intensity is higher than the intensity of the target pixel are excluded due to the pre-knowledge that removing the diffuse reflection never increase the intensity.

With the roughly estimated reflectance component, M_i, the illumination component can be obtained by subtracting the reflectance component from the input image.

3.3 Scene Geometry Construction

The position of the vertex in the 3D scene to which a pixel is projected back can be computed by the position of the pixel in 2D image space as long as we know a triple of points' y-coordinate in 2D image space where one point of them is the centroid of the other two in 3D scene space. Fig.3 illustrates the relationship of the triple of points. By assuming that the point P_c is located at the origin and that the camera is facing +z, the z-coordinate of the corresponding vertex of pixel p in 3D scene space is computed by

$$P_z = (-1 - \mathrm{sig}(\hat{y_p})\sqrt{D})/2A \tag{1}$$

where

$$D = 1 + 4A\Big((s_b^2 + s_t^2)/(s_b^2 - s_t^2) + q\Big),$$

$$A = q + \Big(2/(s_b^2 - s_t^2)\Big), \qquad s_b = \hat{y_p}/\hat{y_2},$$

$$q = \tfrac{1}{2}\Big((\hat{y_1} - \hat{y_2})/(\hat{y_1} + \hat{y_2})\Big), \quad s_t = \hat{y_p}/\hat{y_1},$$

$$\hat{y_k} = |y_k - y_c|, k \in \{p, 1, 2\},$$

and y_k denotes the y position of pixel p_k in 2D image space and sig(·) is the sign operator. The x-coordinate of the corresponding vertex is

$$P_x = (1 + 2qP_z)(x_p - x_c) \tag{2}$$

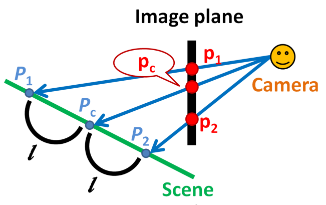

Figure 3: The relationship of the triple of points for creation of the scene geometry.

where x_p and x_c denote the x position of the pixel p and p_c in 2D image space, respectively. Combining Eq.1 and 2, we obtain our 2D-to-3D transformation function f for the pixels on the ground plane:

$$f_g(p, y_c) = [P_x, 0, P_z]^T \tag{3}$$

Nevertheless, the pixels on the wall need another different transformation function. To simplify the computation, we assume that the walls are perpendicular to the ground and that the ground is always beneath the wall. This assumption gives a strong constraint that the x and z-coordinate of the corresponding vertex of a pixel on the wall are same as that of the base pixel, which is the pixel right beneath and is on the edge between the wall and the ground (Fig.4). Therefore, we know that the pixels above the edge of the wall specified by the user are on the wall and the x and z-coordinate of their corresponding vertex in the 3D scene are determined by their base pixels. Their y-coordinate in 3D scene space can be obtained by a similar equation with Eq.2:

$$P_y = (1 + 2qP_z)\delta y_p \tag{4}$$

where δy_p denotes the distance in 2D image space from the pixel p to its corresponding pixel on the edge of the wall. We define the 2D-to-3D transformation function f_w for the pixels on the wall as

$$f_w(p, y_c) = [P_{bx}, P_y, P_{bz}]^T \tag{5}$$

where P_{bx} and P_{bz} denote the x and z-coordinate of p's base pixel's corresponding vertex in the 3D scene.

However, there is still a scaling ambiguity existing in the x and y-coordinate, so we combine the Eq.3 and Eq.5 and rewrite them to obtain our final 2D-to-3D transformation function f:

$$f(p, y_c) = \begin{cases} [\alpha P_{bx}, \alpha P_y, P_{bz}]^T, & \textbf{if } p \textbf{ is on the wall.} \\ [\alpha P_x, 0, P_Z]^T & \textbf{otherwise.} \end{cases} \tag{6}$$

Fig.5 shows the difference after multiplying with α.

3.4 Estimating Scene Geometry

In our implement, we choose the pixels which have the largest and the least y-coordinate in the user specified region in 2D image space as p_1 and p_2, respectively, and y_c, the y-coordinate of p_c in 2D image space, is estimated by minimizing the objective:

$$\operatorname*{arg\,min}_{y_c,\sigma,w} \sum_{p \in \mathbb{R}} \left(\mathrm{Int}\Big(M_i(p)\Big) - w\mathrm{G}\Big(\mathrm{dist}(p, m), \sigma\Big) \right)^2 \tag{7}$$

where

$$\mathrm{dist}(p, m) = \mathrm{proj}(f(p, y_c)) - \mathrm{proj}(f(m, y_c)),$$

$\mathbb{R}$ denotes the region specified by the user, $\mathrm{Int}(\cdot)$ returns the intensity of the given color, $\mathrm{G}(\cdot, \sigma)$ implies a 2D Gaussian function with standard deviation σ, and $\mathrm{proj}(\cdot)$ projects the given vertex onto the plane on which the diffuse reflection shows.

Notice that σ is a two-dimensional vector and the quotient of dividing the first element by the second element indicates the value of α.

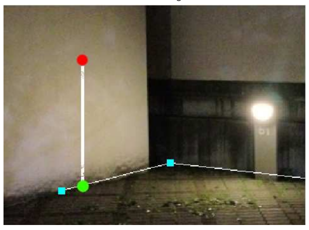

Figure 4: The x and z coordinate of the red pixel's corresponding vertex in the 3D scene is same as that of its base pixel (the green pixel).

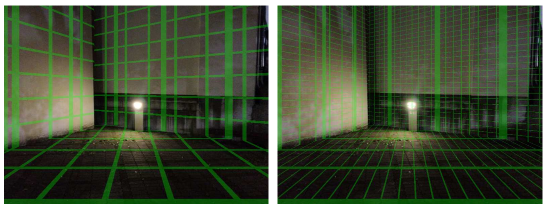

Figure 5: The grid of the geometry generated by our system with multiplying α (left) and without multiplying α (right).

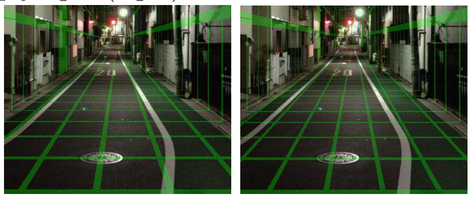

Figure 6: The estimated geometry (left) and the geometry assigned manually (right).

4 Results

Fig.6 shows the geometry estimated by our system(left) and the geometry created manually(right), from which we can see that our system can estimate the geometry almost as good as human's eyes do. The processor of our experiment machine is Intel i7 3.4GHz and it equips GeForce GTX660. The resolution of the input image is about 800x600 and the computation time is around 1 second.

5 Conclusion

We proposed a simple method to estimate the geometry of the scene in an image with requirements of only a small amount of user input. Our method can be applied to the photos in which at least one diffuse reflection is observable.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. In *ACM SIGGRAPH 2009 papers*, pages 24:1–24:11.

[2] Varsha Hedau, Derek Hoiem, and David A. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV 2009*, pages 1849–1856.

[3] Satoshi Iizuka, Yoshihiro Kanamori, Jun Mitani, and Yukio Fukui. An efficient modeling system for 3d scenes from a single image. *IEEE Computer Graphics and Applications*, 99(PrePrints), 2011.