

# Single-View 3D Reconstruction by Learning 3D Game Scenes

Okabe, Makoto  
The University of Electro-Communications | JST CREST

<https://hdl.handle.net/2324/1430812>

---

出版情報 : MI lecture note series. 50, pp.39-42, 2013-10-21. 九州大学マス・フォア・インダストリ  
研究所  
バージョン :  
権利関係 :



# Single-view 3D Reconstruction by Learning 3D Game Scenes

**Makoto OKABE**

The University of Electro-Communications/JST CREST, Tokyo, Japan.

(joint work with Ken Anjyo and Rikio Onai)

## 1. INTRODUCTION

Since the 3D depth information is an important cue for scene understanding, the 3D reconstruction of a scene has been one of the most active research fields in computer graphics. While the multi-view stereo and the depth sensor like Microsoft Kinect have become powerful tools for precisely reconstructing the 3D scene recently, the single-view reconstruction is also a popular: while it is still difficult to create the high-quality result compared with the methods described above, since only a single 2D image is required, it is always easy and intuitive for us to use the method. Also, the resulting animation of touring into a picture always has the visual impact [2, 1].

We propose a novel data-driven approach for single-view reconstruction. We first investigate how 2D image corners or lines appear at the specific 3D surface normal. As the training data, since we want clean 3D scenes, we gather a set of surface normal maps from a consumer 3D game. We apply our corner or line detector to each surface normal map and learn the knowledge. Given an input image, we apply the same corner and line detector to it, estimate the 3D surface normal map using the learned knowledge. We finally integrate the surface normal map to obtain the depth map. We demonstrate that our method is good at reconstructing the rough 3D shape of a single image consisting of man-made objects.

## 2. OUR TRAINING DATASET

We choose a set of surface normal maps of 3D game scenes as our training data (Fig. 1-a), because a 3D game provides a variety of scenes, and also a game scene usually consists of low-resolution polygons, which enables to observe sharp corners and lines easily. To capture 3D scenes from a game, we use the software of Microsoft DirectX: during the game play, we can capture all the information on the graphics card by pushing the print-screen key. We play Anno 1404 produced by Ubisoft [14], and capture 236 scenes. This information includes the 3D depth map, from which we compute the surface normal map. We detect the corners and lines over each of these maps, and learn the correlation of the detected corner or line to the underlying surface normal (Figs. 1-b and c).

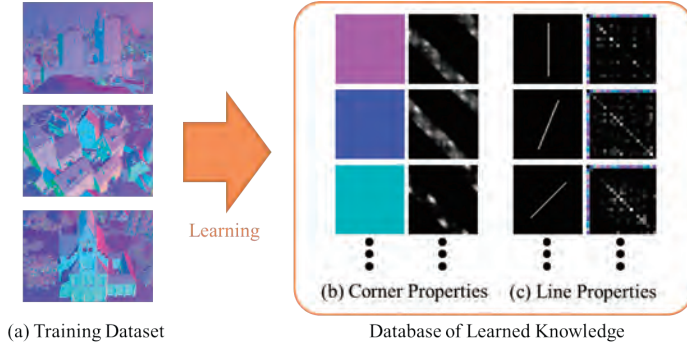


FIGURE 1. Overview of our learning process. In the surface normal map, the pixel color  $(r, g, b)$  is determined proportionally to the surface normal  $(n_x, n_y, n_z)$ .

### 3. LEARNING CORRELATION BETWEEN IMAGE FEATURES AND SURFACE NORMAL

Our corner representation is shown in Fig. 2. Given an input image of a Y-shaped corner as Fig. 2-a, we expect that the three corners of red, yellow, and blue colors would be detected around the Y-junction (Fig. 2-b). In our definition, a single corner consists of the two arms, i.e., a single corner can be defined by a pair of angles,  $\theta$  and  $\phi$  as shown in Fig. 2-c.

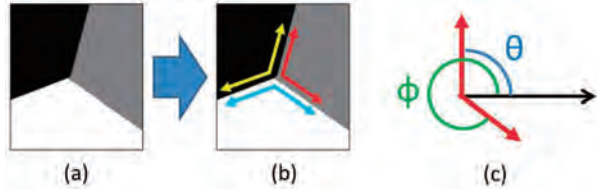


FIGURE 2. Our representation of a corner.

We apply our corner detector to a surface normal map (Fig. 1-a) and learn the correlation between the shape of the detected corner and the underlying surface normal. At each detected corner, we observe the surface normal at each pixel between the arms of the corner. Fig. 3-a shows the appearance frequency of the detected corners that have the green surface normal between their arms. In Fig. 3-a, the horizontal and vertical axes represent the angles,  $\theta$  and  $\phi$ . The pixel intensity is proportional to the appearance frequency of each corner. For example, the pixels inside the red circle of Fig. 3-a have high intensities. One of the corners of these  $\theta$  and  $\phi$  is shown as the red corner in Fig. 3-b, and actually, corners like this appeared a lot with the green surface

normal through our training surface normal maps. Similarly, one of the corners in the yellow circle of Fig. 3-c is shown as the yellow corner in Fig. 3-b, and this type of the corner often appears along with the pink surface normal.

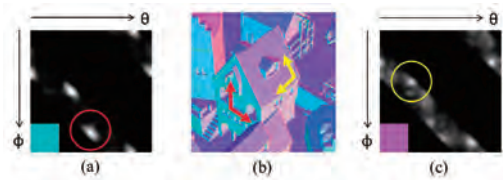


FIGURE 3. We measure the frequency of co-occurrence of a corner and a surface normal.

We also detect lines in a surface normal map, and learn the correlation between the detected line and the pair of surface normals across the line. Fig. 4-a shows the appearance frequencies of the veritically straight line associated with each pair of surface normals. For example, the pixels pointed by the red and yellow arrows in Fig. 4-a have high intensities. These mean that the pair of green and pink normals or the pair of pink and green normals appeared a lot through our training data. As shown as the red and yellow straight lines in Fig. 4-b, such pairs of surface normals often appear. Fig. 4-c shows that the pair of purple and pink surface normals also appear a lot across the diagonal line like the orange line of Fig. 4-b.

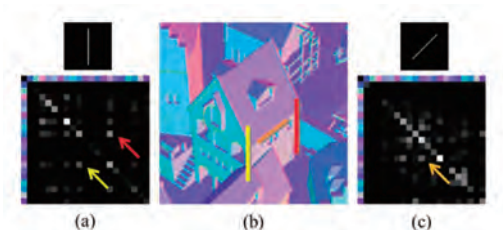


FIGURE 4. Our representation of a corner.

#### 4. 3D RECONSTRUCTION BY LEARNED KNOWLEDGE

Our 3D reconstruction process takes the single image as input (Fig. 5-a), and applies the same corner and line detectors to it (Fig. 5-b). We assume that the detected corners and lines would have the surface normal around them along with the appearance frequencies learned above. We formulate the assignment of surface normals using Markov Random Field (Fig. 5-c), which can be solved efficiently by the graph-cut algorithm. We finally integrate the estimated surface normal map to reconstruct the 3D geometry of the scene (Fig. 5-d).

Fig. 6 shows the surface normal maps. In the first row, we could obtain relatively beautiful surface normal map for the simple synthesized image. In the second and third rows, we could obtain surface normal maps but they are rough. This roughness are caused by the wrong detection of corners and lines. The investigation of more precise detectors of corners and lines would be our future work.

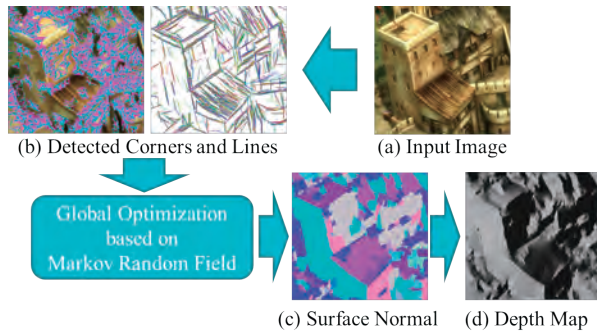


FIGURE 5. The overview of online 3D reconstruction process.

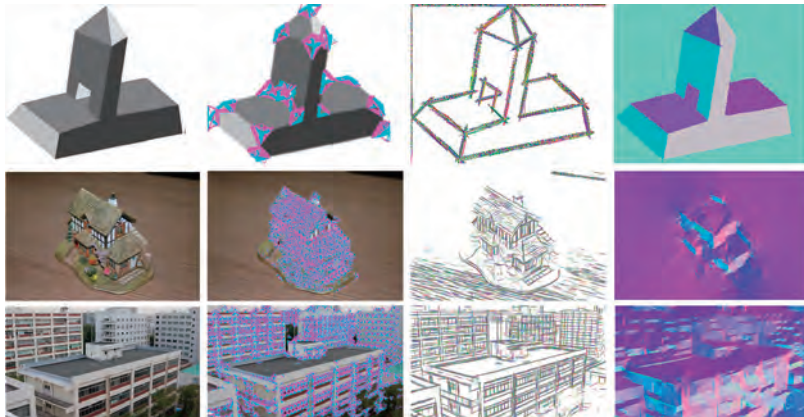


FIGURE 6. From the left: the input image, detected corners, detected lines, and the estimated surface normal.

#### REFERENCES

- [1] D. Hoiem, A. A. Efros, and M. Hebert: Automatic photo pop-up, *ACM Trans. Graph.*, 3, 24: 577-584, 2005.
- [2] Y. Horry, K. Anjyo, and K. Arai: Tour into the picture: using a spidery mesh interface to make animation from a single image, *Proc. SIGGRAPH '97*: 225-232, 1997.