國立交通大學

多媒體工程研究所

碩士論文

3D 街頭立體藝術影像互動式生成系統

An Interactive System for the Synthesis of 3D Street Art Illusion

研 究 生：謝承翰

指導教授：施仁忠　教授

魏德樂　教授

中華民國 一百零二 年八月

3D街頭立體藝術影像互動式生成系統

An Interactive System for the Synthesis of 3D Street Art Illusion

研究生：謝承翰 Student：Cheng-Han Hsieh

指導教授：施仁忠 Advisor：Prof. Zen-Chung Shih

魏德樂 Prof. Der-Lor Way

國立交通大學

多媒體工程研究所

碩士論文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Multimedia Engineering

August 2013

Hsinchu, Taiwan, Republic of China

中華民國一百零二年八月

3D 街頭立體藝術影像互動式生成系統

研究生：謝承翰　　　　　　　　　　指導教授：施仁忠 教授

魏德樂 教授

國立交通大學多媒體工程研究所

摘　　要

近幾年來，3D 街頭錯視覺藝術深受人們歡迎，我們可以在街上、人行道上、廣場中看到此類藝術作品。同時該藝術也被稱為 3D 立體粉筆畫，這類畫作本身就是以二維平面形式被畫在街道上，但若在某一特定透視角度下去觀賞，卻能看見畫作以 3D 立體樣貌來呈現錯視覺之效果。在此，我們提出一個互動式系統，將 CG 模型場景置入現實照片影像中來模擬 3D 街頭藝術，使用者無需真的到該場景中去作畫。首先我們透過相機校正來估測該場景中，相機當時拍攝的位置；接著利用現有的 NPR 技術，來給予我們 CG 模型圖增添藝術風格化。最後使用 Poisson Blending 將風格化後之模型圖與原始照片場景圖合併，此方法同時能將場景地面材質細節反映在模型圖上。透過簡單的互動調整，使用者能得到非常生動有趣且相似於真實 3D 街頭立體藝術之結果圖。

An Interactive System for the Synthesis of 3D Street Art Illusion

Student: Cheng-Han Hsieh Advisor: Prof. Zen-Chung Shih

Prof. Der-Lor Way

Institute of Multimedia Engineering

National Chiao-Tung University

ABSTRACT

3D Street Art has become more popular in recent years. Many 3D illusion artworks are shown on pavement such as streets, sidewalks, and town squares. It's often known as 3D chalk art which a 2D artwork drawn on the street itself giving the viewer a 3D optical illusion from a certain perspective. This thesis presents an editing system to synthesis computer graphic models into a realistic photograph without requiring access to the scene to draw 3D street paintings on the ground. First the camera position was recovered using camera calibration algorithm. Second, a NPR stylization technique was applied to create artistic image of 3D models with the same camera position. Finally, the NPR image was composited into source photograph using Poisson approach. All enhanced pavement texture details were also blended on object image. With a small amount of specifications and adjustments, user can create a lively and interesting result which likes those realistic 3D art illusions.

Acknowledgements

First of all, I would like to express my sincere gratitude to my advisors, Prof. Zen-Chung Shih and Prof. Der-Lor Way for their guidance and patience. Without their encouragement, I would not complete this thesis. Thanks also to all the members in Computer Graphics and Virtual Reality Laboratory for their reinforcements and suggestions. Thanks for those people who have supported me during these months. Finally, I want to dedicate the achievement of this work to my family.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

There are many illusions which deceive the human visual system into perceiving something that does not exist, or incorrectly perceiving what is present. In general, literal optical illusion is characterized by the difference between the sight seen by eyes and the actual physical objects that make the image. Some artists use this characteristic to create interesting visual illusions by extending it to 3D. For example, tow lines with different lengths originally, but when changing our viewing direction, it might look like they both have the same length.

Street art, also commonly known as street painting, or sidewalk art, is a performance art of rendering artistic designs on pavement in public space. Street artists do not aspire to change the definition of an artwork, but rather to question the existing environment with its own language. This art have been around since the sixteenth century when Italian Renaissance *Madonnari* painters created stunning murals to decorate the interior walls of luxurious villas or painted religious pictures directly on the paved public squares. When it comes to 3D, it's often a matter of perspective, the magic of 3D is created by painting a 2D artwork and requiring viewer to stand at a certain vantage point to capture the right perspective. This process is called *anamorphosis*, painting appears flat and distorted, but from a specific angle it comes alive. As shown in Figure 1.1, street artists usually design their artworks by creating compositions that appear to rise from, or fall into the ground. Sometimes, it's also designed to interact with human or the environments to make the illusion become more vivid. 3D street art can be very interesting and breathtaking. Viewers are often impressed easily since these artworks show excellent optical illusions to make people believe it is real.

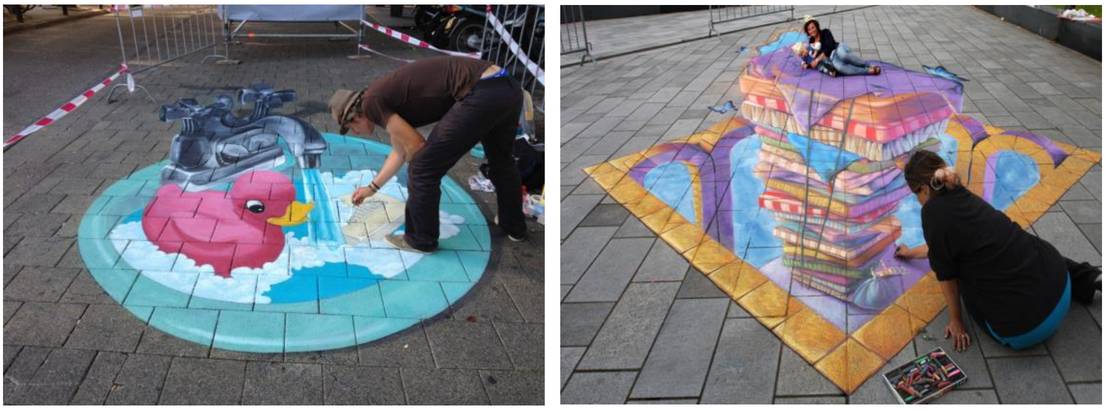


Figure 1.1: Real 3D street paintings by artists. Copyright by *We Talk Chalk Studios*, all rights reserved [32, 33].

However, drawing such street paintings is not a trivial task especially for non-artist. It requires accurate measurements of the scene information before drawing the distorted painting, and advanced techniques for conveying the artwork. Thus, most such illustrations are only handcrafted by skilled artists. This thesis is inspired from those artworks. Instead of thinking how to deform a painting to anamorphic illusion, user can select a 3D model as input which already gives the 3D-like effect and then inserts it in a street photo. We propose an editing system to create the illusion which imitates real 3D street art without requiring drawing painting on the pavement.

Figure 1.2 describes an overview of our 3D Street Art Illusion system architecture. First, user needs to annotate two pairs of parallel lines in the street image. Then two vanishing points were estimated for recovering the camera pose automatically. The 3D model was rendered with the same camera position to get an object image. Since the original color on model’s surface is too smooth and humdrum, it lacks of artist’s painting style. Hence a non-photorealistic rendering (NPR) technique was applied to the object image to create different stylization. Finally, embedding the NPR image in source street photo is done by using modified Poisson blending approach [23]. Several experimental results were demonstrated, showing that it’s potentially a useful and creative tool for user.

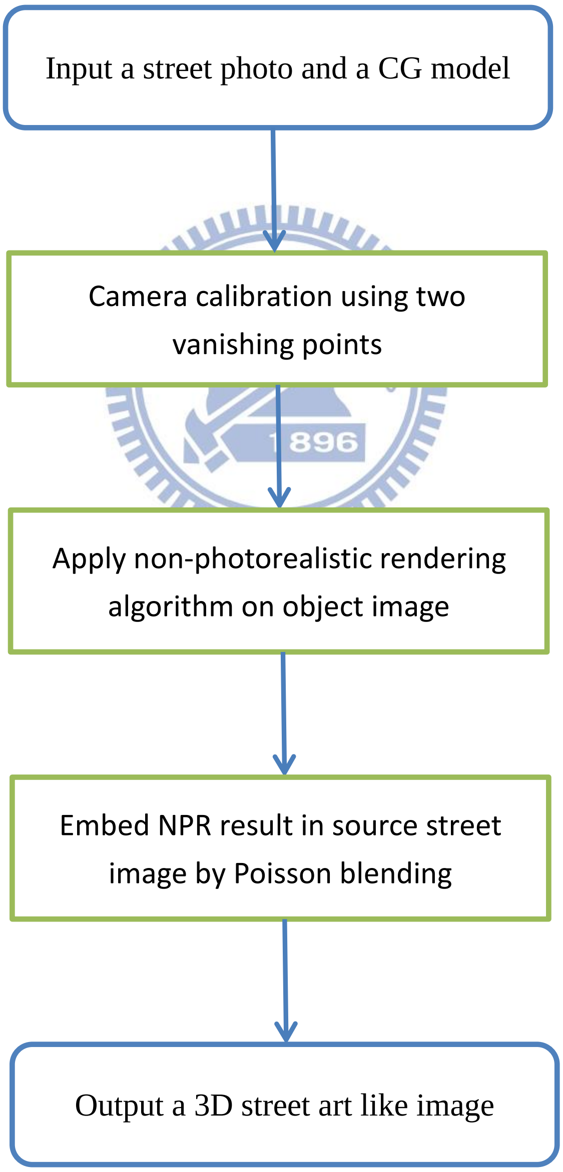

Input a street photo and a CG model
Camera calibration using two vanishing points
Apply non-photorealistic rendering algorithm on object image
Embed NPR result in source street image by Poisson blending
Output a 3D street art like image

Figure 1.2: Flow chart of our system architecture.

The rest of this thesis is organized as follow. Chapter 2 reviews some related works. Then, Chapter 3 described the major method and algorithm that we used in each step, such as camera calibration, stylized technique, and Poisson image blending. Chapter 4 shows many interesting experimental results. Finally, some conclusions and future works were discussed in Chapter 5.

Chapter 2
Related Works

2.1 Camera Calibration

Camera calibration is indispensable to extract metric information from 2D images. It is the basic premise of many computer vision issues like metrology, 3D reconstruction, augmented reality and visual surveillance. Calibrating a camera consists of determining its intrinsic and extrinsic parameters. Some previous works need to put a reference object in the scene, and then collect a set of correspondences between 3D reference points on object and their projections on image plane to solve this problem. Tsai et al. [25] made use of a 3D known metric structure and undergo a precisely known translation to obtain constraints. Zhang et al. [30] further achieved calibration by observing a 2D planar pattern shown at a few (at least two) different orientations without knowing the motion. However, such reference-object-based calibrations are inconvenient for our approach, since it always needs to place an apparatus in the scene before taking a picture. Additionally, calibrated template is not suitable in a wide scene; the projection of object is too small to get accurate calibration.

Hence, self-calibration method becomes more essential recently which doesn’t use any calibration object. *Vanishing-point-based* calibration techniques have been devised. Caprile and Torre [2] firstly put forward the idea of using vanishing point for self-calibration. They solved intrinsic parameters from a single camera but extrinsic parameters from multi-cameras. Cipolla et al. [3] described a method to use three vanishing points and two reference points from two viewing angles to recover projection parameters. In order to minimize all these requirements, we calibrate the street image based on Guillou's approach [8] which only needs two vanishing points contained in one image.

2.2 Integrating Synthetic Objects into photos

Many applications require user to insert synthetic objects into images. Several researches focus on how to make objects look more real. Thus estimating light information of the scene is essential for rendering. Cao and Shag [4] recovered the light source orientation from two different views of a scene given with two vertical lines and their cast shadows. Debevec et al. [6] and Abad et al. [1] presented a method that used a light probe to capture a physically accurate radiance map at the location of the synthetic objects. Then it synthetically illuminates new objects with reflectance characteristic. Lopez-Moreno et al. [15] proposed another approach that use any object in the image as a virtual light probe. Firstly it predicts the azimuth angle of the light along object's silhouette. Next it uses the globally convex assumption to find the zenith angle. Finally it estimates the ambient light intensity by analyzing median intensity in the shadow area. Estimating light sources automatically from a single image is a difficult task. Instead, Karsch et al. [14] described a method that asks user to annotate the light source at the beginning. Then it seeks appropriate light parameters to minimize the difference of each pixel value between rendered image and original image. These works all try to enhance the realism of inserted object in the scene. Conversely, Fischer et al. [7] applied stylization techniques for reducing the visual realism of both camera image and virtual objects in real time. Thus the synthetic object seems to be a part of the real scene visually.

2.3 Image-Based Non-Photorealistic Rendering

The color representation in computer graphic is too smooth and humdrum to be represented as an artist's painting. Thus we apply rendering technique to our object image. Lots of non-photorealistic rendering (NPR) algorithms in painterly rendering have demonstrated that certain artistic styles can be mimicked. One of the fundamental issues in NPR is how to convert the source images into the corresponding paintings with the desired artistic effects.

Some previous methods focus on stroke-based rendering, such as water color, oil painting, impression style, and etc [11, 16, 17, 22, 26, 31]. They apply strokes to image based according to various orientations. Hertzmann [12] created a hand-painted style image built up in a series of layers with various radiuses of circular brush. He also proposed a system [13] that uses a height map which is assigned to each stroke. Then an embossing painting result is rendered by bump-mapping the image's colors with the height map. Hays et al. [9] expanded Hertzmann's work and used radial basis functions to globally orient brush strokes across time and space, which allowed for highly temporally coherent video based animations. Their experimental results resembled the style of Van Gogh. Rather than defining stroke alignment fields globally, Olsen [21] simulated several kinds of field model for each region of image to create the result in an expressionist style. Zeng et al. [29] first decomposed input image into a hierarchy of components in a parse tree. They chose appropriate brushes for every component and layer to enrich image semantic information.

Another popular style of artistic painting is the pencil drawing. Sousa and Buchanan [24] developed an interactive system for pencil drawing from images by employing the blender and eraser model. Lu et al. [18] proposed a parametric model to fit tone distributions of input natural image for the pencil sketch by artists and combined it with line drawing to create final result. Some researches [20][19] have employed Line Integral Convolution (LIC) to produce desired pencil strokes. Yamamoto et al. [27] divided source image into different intensity layers without using image segmentation and applied LIC filter to each layer, then added them together to obtain the final pencil drawing. Yamamoto [28] further extended the LIC technique to generate the colored pencil drawing. They reproduced color image from duotone color to select the best color set automatically for each region, and used the KM model for performing the optical blending of the overlaid layers. Hata et al. [10] presented a novel technique for converting input image into pencil drawing with such emphasis and elimination effect using a saliency map. This thesis modified Mao's technique [21] to provide visible line strokes to enhance object image with artistic painting style.

Chapter 3
The Algorithm

3.1 Camera Calibration

Camera calibration is a necessary step in 3D computer vision. It is usually used in many applications to recover 3D quantitative measures about the observed scene from 2D images. Given a calibrated camera we can easily get the correct perspective view of an object according to its distance from the camera. Many traditional methods place calibrated templates in the scene, and compute the projection matrix by using several pairs of points which are on the templates. Although these techniques could get highly accurate parameters of the camera, they are still restricted to many applications. Since user would not always want to place any apparatus before taking a picture of the scene. Even so, the projections of template are too small to make estimation accurate when it is put in a wide area. In order to improve the convenience of our system, we focus on self-calibration without using any template.

The most common camera model is pin-hole camera which projects 3D Euclidean space onto 2D image plane. With assumption of perfect projection, the collinearity of points is preserved. It means every line in the real scene is projected as a line on image plane too. An interesting property is found that a set of parallel lines in Euclidean space would intersect at a point on image plane through the projective transformation. This point is known as the *vanishing point*. Once two vanishing points lie on the same plane in the scene, it can define a line in image which is called the *vanishing line*. In Figure 3.1, by assuming the green lines and the red lines are both parallel lines, it shows that two vanishing points and one vanishing line could be derived from a rectangle. Thus we see a vanishing point is defined by a direction generally.

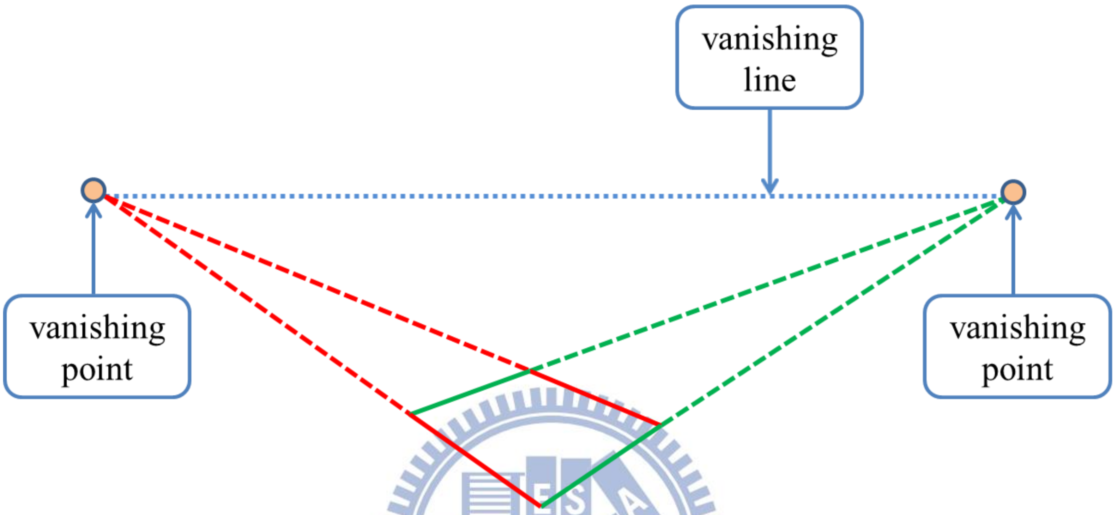


Figure 3.1: Two vanishing points and one vanishing line.

In order to create a 3D street art image, the input scene is usually taken of man-made environment which is full of many parallel lines and orthogonal edges, such as street, doors, windows, or pavement tiles. Before estimating the camera pose, user can annotate two pairs of parallel lines that are mutually orthogonal directions of such an environment is presented, then intersect these lines in points. Our camera calibration method is based on Guillou's approach [8] which only needs two finite vanishing points in one image.

For a pin-hole camera model, the perspective transformation is used to denote a mapping from 3D world space to 2D image plane. It can be represented in homogeneous coordinates by a 3×4 projection matrix, **P**:

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{1}$$

$(X_i, Y_i, Z_i, 1)^T$ are homogenous coordinates of the world space point, and $(u_i, v_i, 1)^T$ are homogenous coordinates of the projected point on image plane. λ_i is a scaling factor. The projection matrix **P** can be decomposed into the 3×3 rotation matrix **R**, the 3×1 translation vector **T** and the 3×3 camera intrinsic matrix **K**:

$$\mathbf{P} = \mathbf{K}\ [\ \mathbf{R}\ \ \mathbf{T}\ ] \tag{2}$$

$$\mathbf{K} = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

α_u and α_v represent the focal length f in terms of pixel dimensions in the *u* and *v* directions. *s* is referred to the skew coefficient between two axes. u_0 and v_0 are the pixel coordinates of the principle point which is the intersection of the optical axis with image plane. An assumption is often used by taking skew factor as zero $(s = 0)$ and the aspect ratios to be unit $(\alpha_u = \alpha_v = f)$, so that the degrees of freedom in intrinsic matrix **K** could be reduced from 5 to 3. In general case, the principal point would be located at the coordinates of image center. We only need to compute the focal distance *f*. Thus all the intrinsic parameters would be obtained.

We consider there is a rectangle *ABCD* which lies in the world space, and its perspective projection on the screen plane is a quadrilateral *abcd*, as shown in Figure 3.2. By intersecting two pairs of parallel lines, we get two finite vanishing points, denoted as $V_1 = (v_{1i}, v_{1j})$ and $V_2 = (v_{2i}, v_{2j})$. Let P be the image center, and $\overline{PG}$

is perpendicular to line $\overline{V_1V_2}$, hence $\overline{OG}$ is also perpendicular to $\overline{V_1V_2}$. Obviously triangles OV_1G, OV_2G and OGP are all right triangles, then ΔOV_1G and ΔOV_2G are supposed to be similar. According to the property of similar triangle, we have:

$$\frac{V_1G}{GO} = \frac{OG}{GV_2} \tag{4}$$

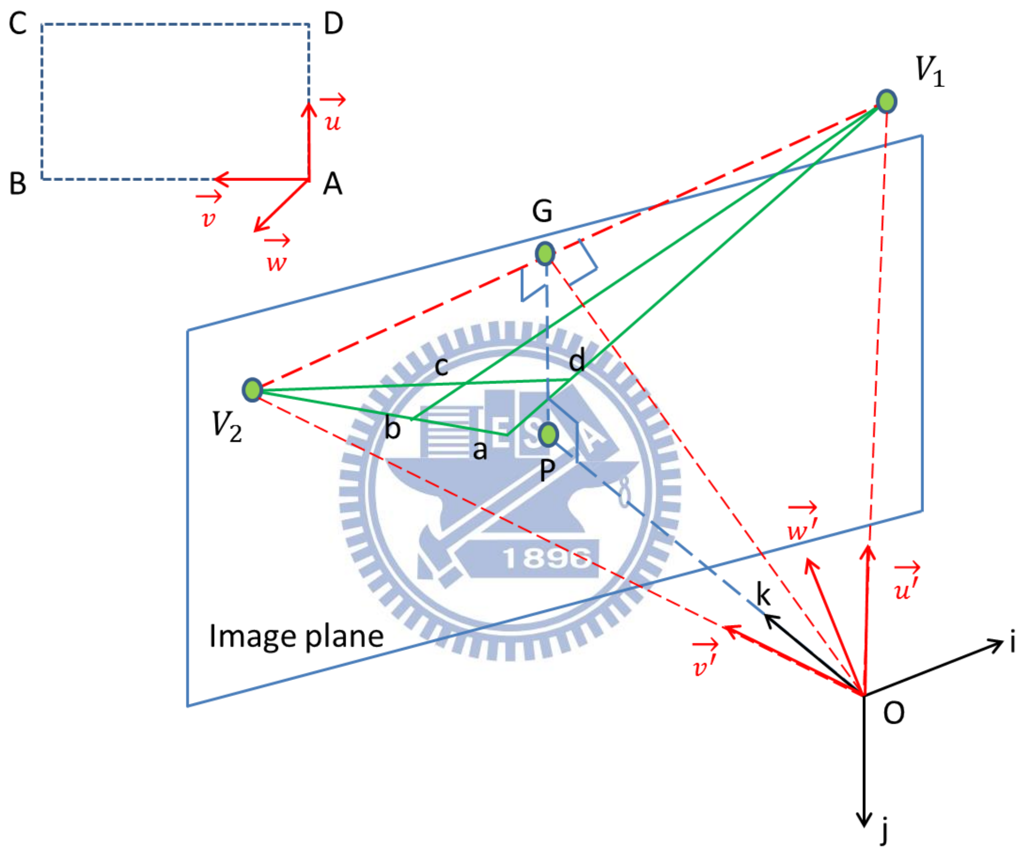


Figure 3.2: The illustration of perspective projection from world space to image.

The camera focal distance *f* equals to the length of $\overline{OP}$ actually, which means that:

$$f = \|OP\| = \sqrt{\|OG\|^2 - \|PG\|^2} \tag{5}$$

$\|PG\|$ is determined once we calculate the coordinates of point *G*. Combining equation (4) and (5), focal length *f* can be further written as:

$$f = \sqrt{\|V_1G\| \times \|GV_2\| - \|PG\|^2} \tag{6}$$

The extrinsic parameter matrix **R** describes a rigid motion between the world and camera coordinate systems, which are represented by $R_w = (A, u, v, w)$ and $R_c = (O, i, j, k)$ respectively. Another coordinate system $R_{c'}$ can be built where the center is also at *O* and has the same orientation as R_w by considering three vectors:

$$\begin{aligned}
\boldsymbol{u}' &= \frac{\overrightarrow{OV_1}}{\|\overrightarrow{OV_1}\|} = \left(\frac{v_{1i}}{\overrightarrow{OV_1}}，\frac{v_{1j}}{\overrightarrow{OV_1}}，\frac{f}{\overrightarrow{OV_1}}\right)^T, \qquad \mathbf{R} \cdot \mathbf{i} = \boldsymbol{u}'\text{；}\\
\boldsymbol{v}' &= \frac{\overrightarrow{OV_2}}{\|\overrightarrow{OV_2}\|} = \left(\frac{v_{2i}}{\overrightarrow{OV_2}}，\frac{v_{2j}}{\overrightarrow{OV_2}}，\frac{f}{\overrightarrow{OV_2}}\right)^T, \qquad \mathbf{R} \cdot \mathbf{j} = \boldsymbol{v}'\text{；}\\
\boldsymbol{w}' &= \left(w_i'，w_j'，w_k'\right) = \boldsymbol{u}' \times \boldsymbol{v}', \qquad \mathbf{R} \cdot \mathbf{k} = \boldsymbol{w}'\text{，}
\end{aligned} \tag{7}$$

Since the focal length f is already determined from equation (6), we know $\|\overrightarrow{OV_1}\| = \sqrt{{v_{1i}}^2 + {v_{1j}}^2 + f^2}$, $\|\overrightarrow{OV_2}\| = \sqrt{{v_{2i}}^2 + {v_{2j}}^2 + f^2}$. Regarding the basic vectors (i, j, k) of R_w as unit vectors, so that $i = (1, 0, 0)^T$, $j = (0, 1, 0)^T$ and $k = (0, 0, 1)^T$. Now we get the rotation matrix **R** by using (8):

$$\mathbf{R} = \begin{bmatrix} \dfrac{v_{1i}}{\sqrt{{v_{1i}}^2 + {v_{1j}}^2 + f^2}} & \dfrac{v_{2i}}{\sqrt{{v_{2i}}^2 + {v_{2j}}^2 + f^2}} & w_i' \\ \dfrac{v_{1j}}{\sqrt{{v_{1i}}^2 + {v_{1j}}^2 + f^2}} & \dfrac{v_{2j}}{\sqrt{{v_{2i}}^2 + {v_{2j}}^2 + f^2}} & w_j' \\ \dfrac{f}{\sqrt{{v_{1i}}^2 + {v_{1j}}^2 + f^2}} & \dfrac{f}{\sqrt{{v_{2i}}^2 + {v_{2j}}^2 + f^2}} & w_k' \end{bmatrix} \tag{8}$$

The final step in camera calibration is to solve the translation vector **T**. The scaling factor λ is not necessary to be calculated in our case. Because the magnitude of each synthetic object is distinct from the others, user can get various perspective sizes of object by adjusting the scaling factor. Assuming a reference point H on image plane corresponding to the origin of world space, then **T** would be:

$$\mathbf{T} = \lambda \frac{OH}{\|OH\|} \tag{9}$$

Once the extrinsic and intrinsic parameter matrices are both found out, we can project the synthetic object of world space to the image plane correctly.

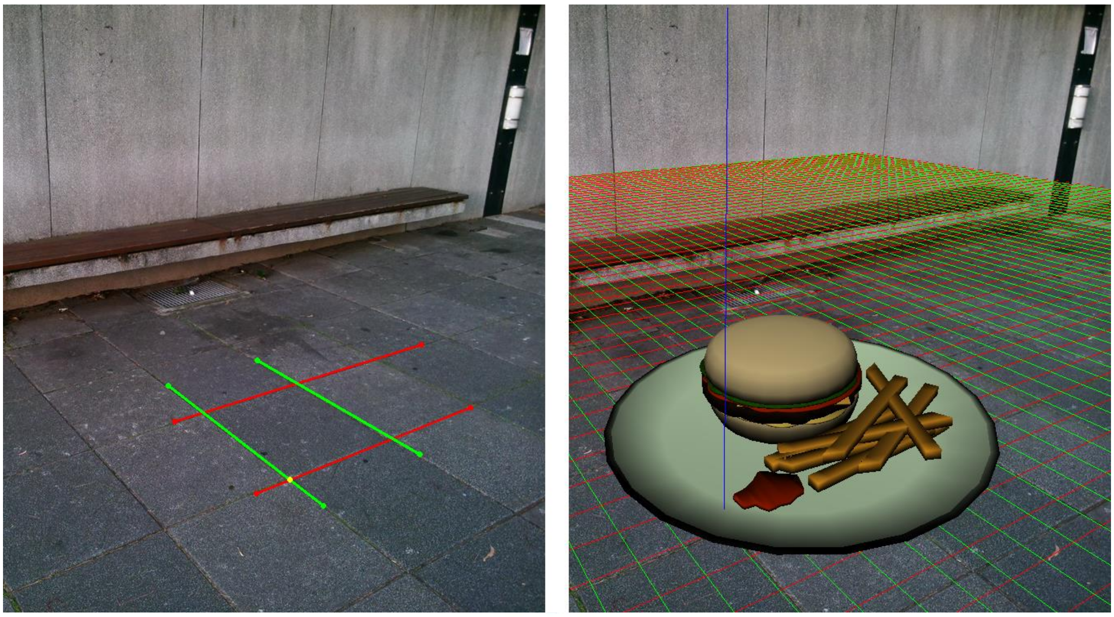


Figure 3.3: (Left) User specifies two pairs of parallel lines in street photo. (Right) Input synthetic object.

3.2 Applying NPR Style to Object Images

As we mentioned at the beginning, the surface colors of synthetic object are too monotonous to be conveyed as an artist’s painting. In general, an actual street artwork is highly related to the media it draws on, such as ground or wall. To put it more concretely, the amount of pigment deposition is affected by the convex or concave part of the media. For a rough plane, the color in each area of the painting would vary its intensity randomly. Another noticeable property of artist’s painting is stroke depiction, which is usually important for representing the shapes and textures of an object. This thesis is based on Mao’s technique [20] to simulate such effects.

In Figure 3.4, the flowchart illustrates the whole framework of our rendering method in the following steps:

(1) Use two intensity thresholds to divide the object image (Figure 3.4(a)) into three layers (Figure 3.4(d)).

(2) Create white noise (Figure 3.4(e)) for each layer.

(3) Segment input image (Figure 3.4(a)) into several regions (Figure 3.4(b)).

(4) Detect detailed edges (Figure 3.4(c)) of the object and compute each pixel’s local direction.

(5) Generate a vector field (Figure 3.4(f)) to define the stroke orientations.

(6) Apply a low-pass filter to the white noise image (Figure 3.4(e)) along the local streamlines of the vector field (Figure 3.4(f)).

(7) Composite the resulting image with texture sample (Figure 3.4(g)), and enhance the outlines to obtain the final drawing (Figure 3.4(h)).

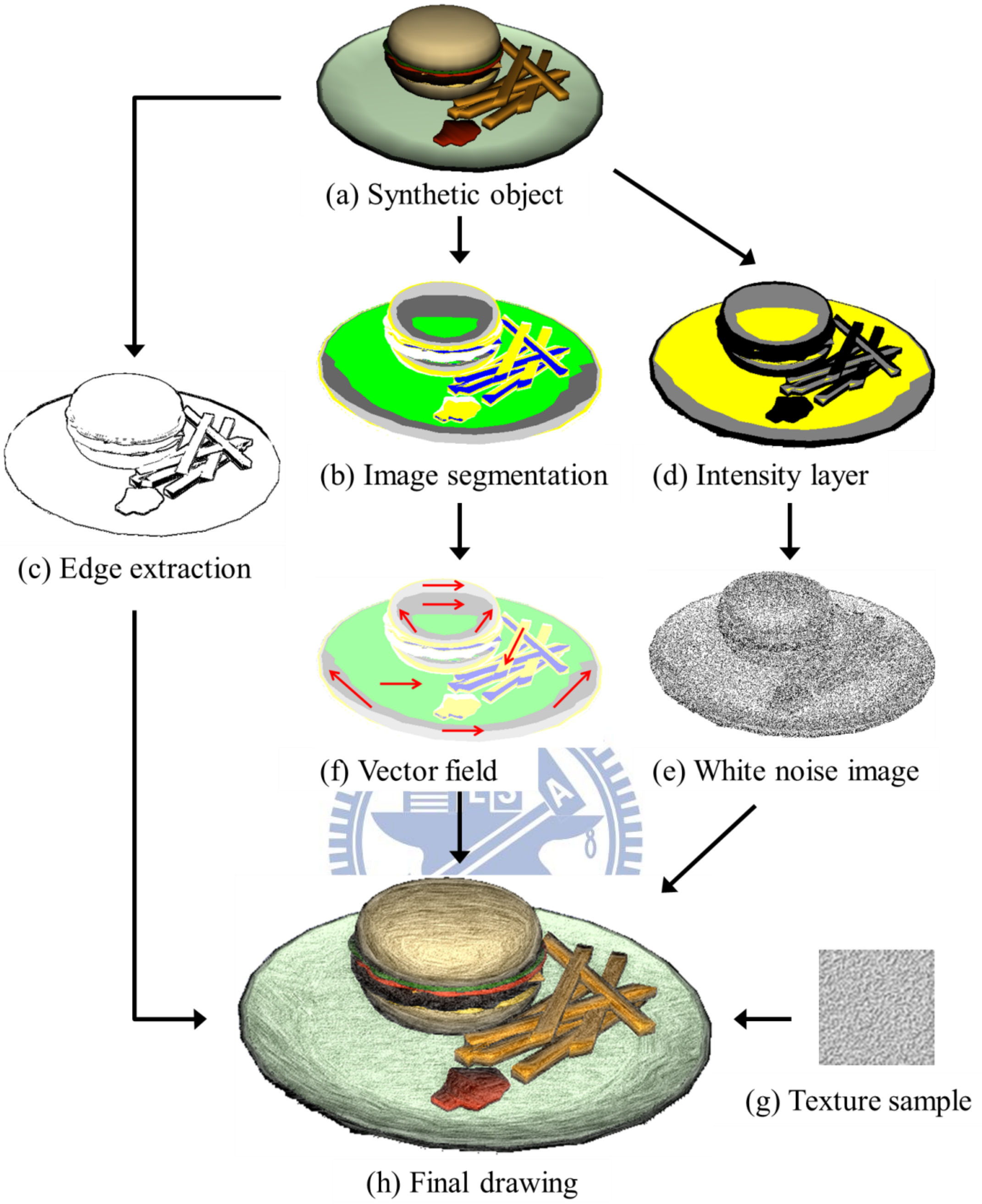


Figure 3.4: The framework of rendering method.

Given an input image, we first convert its RGB signals to YCrCb space and denote each pixel's intensity by I_{input}. The gray-scale tone of the resulting drawing is mainly decided by a white noise image. Thus the tone of input image is used to guide the distribution of noise. The original method described in previous work dealt with this process according to a uniform criterion. However, once the intensity in each area is too high or too low, it would be difficult to generate visible strokes in that region. To solve this problem, we divide the input image into three layers from brightness to darkness. The intensity I_{noise} of corresponding pixel is decided by the following way:

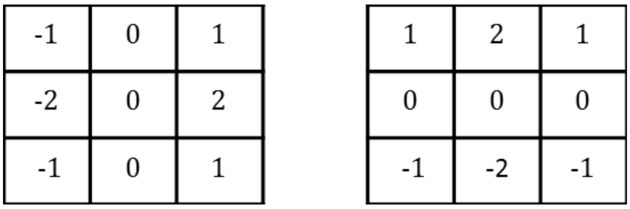
$$I_{noise} = \begin{cases} if\ I_{input} \geq\ H_1, & I_{noise1} = \begin{cases} I_{max1}:\ \ if(P > T_l) \\ I_{min1}:\ \ else \end{cases} \\ \\ if\ H_2 \leq I_{input} <\ H_1, & I_{noise2} = \begin{cases} I_{max2}:\ \ if(P > T_l) \\ I_{min2}:\ \ else \end{cases} \\ \\ if\ I_{input} <\ H_2, & I_{noise3} = \begin{cases} I_{max3}:\ \ if(P > T_l) \\ I_{min3}:\ \ else \end{cases} \end{cases} \tag{10}$$

$$H_n = I_{average} + \omega_n \times S, \quad n = 1,2 \tag{11}$$

$$T_l = \varepsilon_l \left(1 - \frac{I_{input}}{255}\right), \quad l = 1,2,3 \tag{12}$$

Where $I_{average}$ is the average intensity of input image, and S is the standard deviation value. Two thresholds H_1 and H_2 can be adjusted by changing the parameters ω_1 and ω_2 respectively. Let P be a random number which is normalized in the range [0, 1]. ε_l is the coefficient for controlling the tone of each layer. Higher value of ε_l will increase the chance of adding more noise. I_{max} and I_{min} are the grey values of the output pixels. While the difference between I_{max} and I_{min} increases largely, the stroke will become sharper.

Stroke is an important factor contributing to the impression of an artwork. Artists usually draw strokes to convey the shades, shapes and textures of objects. If a region contains visible directional texture, then the strokes should match the same orientation of that texture. Here we segment the input image into different regions R_k using k-means clustering, and apply Sobel operator (Figure 3.5) to generate the vector field:



-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Figure 3.5: Sobel convolution kernels.

The two 3×3 kernels are convolved with the original image to calculate the measurements of the gradient component in horizontal (G_x) and vertical (G_y) directions respectively. In discrete case, the gradient magnitude $|G|$ and the local edge direction θ are given by:

$$|G| = \sqrt{(G_x)^2 + (G_y)^2} \tag{13}$$

$$\theta = \arctan\left(G_y/G_x\right) + \pi/2 \tag{14}$$

Where θ_j is the average angle of region R_j, $j = 1,2, \dots k$. Let $G_{j,n}$ be the gradient value of the nth pixel in region R_j, $\theta_{j,n}$ denotes the local vector of the same location, $n = 1,2, \dots N_j$. Rather than specifying a direction for each region, we use a threshold t to subdivide the area into two parts. Those positions with high gradient values should maintain its own orientations, and the smooth parts would follow another uniform direction which can also be defined by user. The final vector $V\theta_{j,n}$ of each pixel is obtained through a criterion as below:

$$V\theta_{j,n} = \begin{cases} \theta_{j,n}, & if \quad G_{j,n} > t \\ \theta_j, & if \quad G_{j,n} \leq t \end{cases} \tag{15}$$

The outline extraction is accomplished by determining whether the gradient magnitude is large enough. Given lower intensity values to the outlines, the contours of objects would become more visible.

Once the vector field and white noise image are already generated, a low-pass filter is applied to smear out the white noise image in the direction of vector filed. Unlike the original Line Integral Convolution (LIC) technique proposed by Cabral et al. [5], we assume that the local vector field can be approximated by a straight line. Sometimes, when artists try to draw the strokes along the region which has higher curvatures, they might use straight lines to depict that region instead of smooth curves. Here we adopt a DDA line drawing technique which is used for linear interpolation of variables over an interval between starting and ending point, and generate the filter kernel tangential to the local vector. To maintain symmetry about a cell, the kernel goes in both positive and negative directions based on a fixed distance. Next the noise image is mapped one-to-one onto the vector field. Pixels under the filter kernel are summed, normalized by the length of the filter kernel. The final output value of each pixel *F(P)* is given by equation (16):

$$F(P) = \sum_{i=-L}^{L} H(P_i) \times \omega_i \Bigg/ \sum_{i=-L}^{L} \omega_i \qquad (16)$$

Denote L as the half length of the stream line. $H(P_i)$ is the noise value of pixel covered by the filter. ω_i is the contribution of P_i to cell point. To avoid excessively blurring the detail textures, the stream line length can be randomly decided according to a criterion. If a pixel has large gradient value, we would decrease its length; otherwise, a long length stroke should be chosen. Combining the LIC image with a texture sample (Figure 3.4(g)), our final rendering result is shown in Figure 3.4(h).

3.3 Embedding Synthetic Object into Street Image

After the non-photorealistic rendering process, the stylized object image is inserted into the source photo, as shown in Figure 3.7(b). If a typical copy-and-paste method were used, the result, as shown in Figure 3.7(a) would be unreasonable since it doesn’t reflect the property of the ground surface. To take it into consideration, we apply Poisson blending approach to generate the final result. Figure 3.6 illustrates the notations of the blending work. *v* is the guidance vector field. And Ω is a closed domain with boundary δΩ, the size of Ω is usually the same as the synthetic object, as shown in Figure 3.7(c). Let *S* be a known scalar function defined over the source image, and let *T* be an unknown scalar function defined over the interior of Ω.

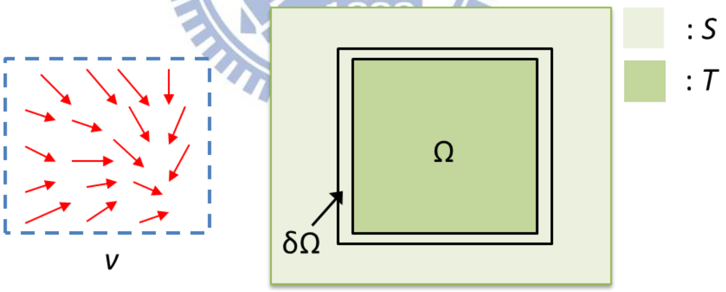


Figure 3.6: The notations of Poisson blending work.

Traditional Poisson image blending can embed the object image into the source image inside the domain Ω seamlessly. Each pixel value *T*(*p*) is calculated by using the following equation (17):

$$|N_p|T(p) - \sum_{q\in N_p\cap\Omega} T(q) = \sum_{q\in N_p\cap\delta\Omega} S(q) + \sum_{q\in N_p} v_{pq} \tag{17}$$

Where N_p are the 4-connected neighboring pixels of p. v_{pq} is the gradient value between pixel p and its neighbor q. Here a mixing gradients type is used, the guidance field v is decided by the gradient fields of both source image and object image. $S(q)$ denotes the value of pixel q from source image.

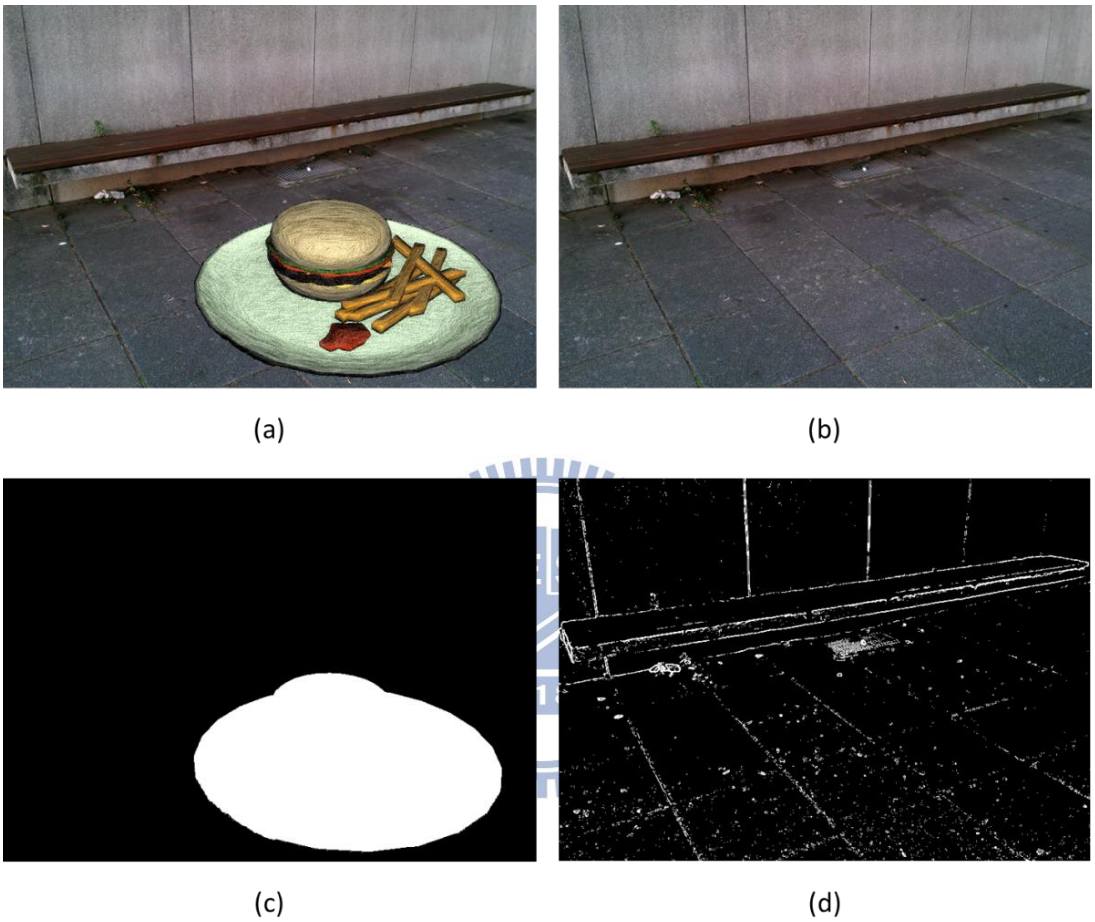
However, the original approach would probably fail to produce a satisfied result. Once the source image contains more high gradient textures, it can barely see the object image. On the other hand, object image might hide some important details of source image due to a group of sharp strokes. In order to solve such cases, a modification of the gradient value v_{pq} determination is used, as below:

$$v_{pq} = \begin{cases} if\ D(p)\ is\ true,\ \ v = \alpha S' \\ else,\ \begin{cases} if\ |S'| > |O'|, \begin{cases} if\ R < Z, \quad v = \beta S' \\ else,\ \ v = (1-\rho)S' + \rho O' \end{cases} \\ if\ |S'| \le |O'|, \quad v = (1-\rho)S' + \rho O' \end{cases} \end{cases} \tag{18}$$

$$S' = S(p) - S(q),$$

$$O' = O(p) - O(q),$$

Where S' and O' are the gradient values of source image and stylized object image respectively. $D(p)$ is used to indicate whether the pixel p of source image is a detail texture or not (Figure 3.7(d)). If it were true, v_{pq} would be definitely decided by S'. R is a random number in the range [0, 1], and Z is a threshold which controls the probability of choosing S' for v_{pq}. In most cases, $|S'|$ might be larger than $|O'|$, thus it is necessary to avoid over blending. For rest conditions, output gradient is composited of S' and O' with a weighting value ρ. Higher value of ρ means it can preserve the original color of object image. Conversely, smaller value of ρ can match the tone with source image closely. Both α and β are parameters which enhance the visibility of source detail textures.



(a)

(b)

(c)

(d)

Figure 3.7: (a) stylized object image, (b) source image, (c) object mask, (d) detail texture map.

As shown in Figure 3.8, the final blending result is similar to the realistic 3D street artwork since it reflects the ground texture successfully. While a painting is drawn on the pavement, it’s not easy to deposit the pigments at the concave parts. The edges between tiles should show up clearly. Another obvious phenomenon is that some ground texture details could be seen above the object image. Generally, artists might only smear the smooth region of painting with a small amount of pigment. Hence the details of ground are not supposed to be hidden.

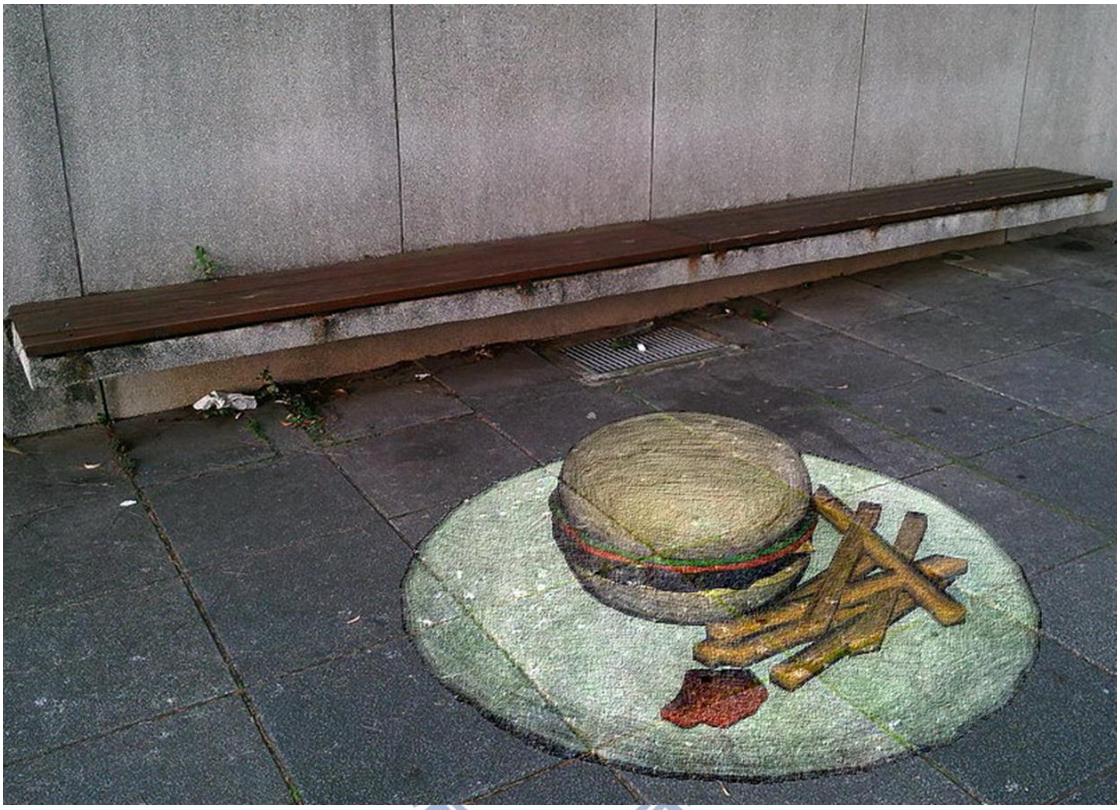


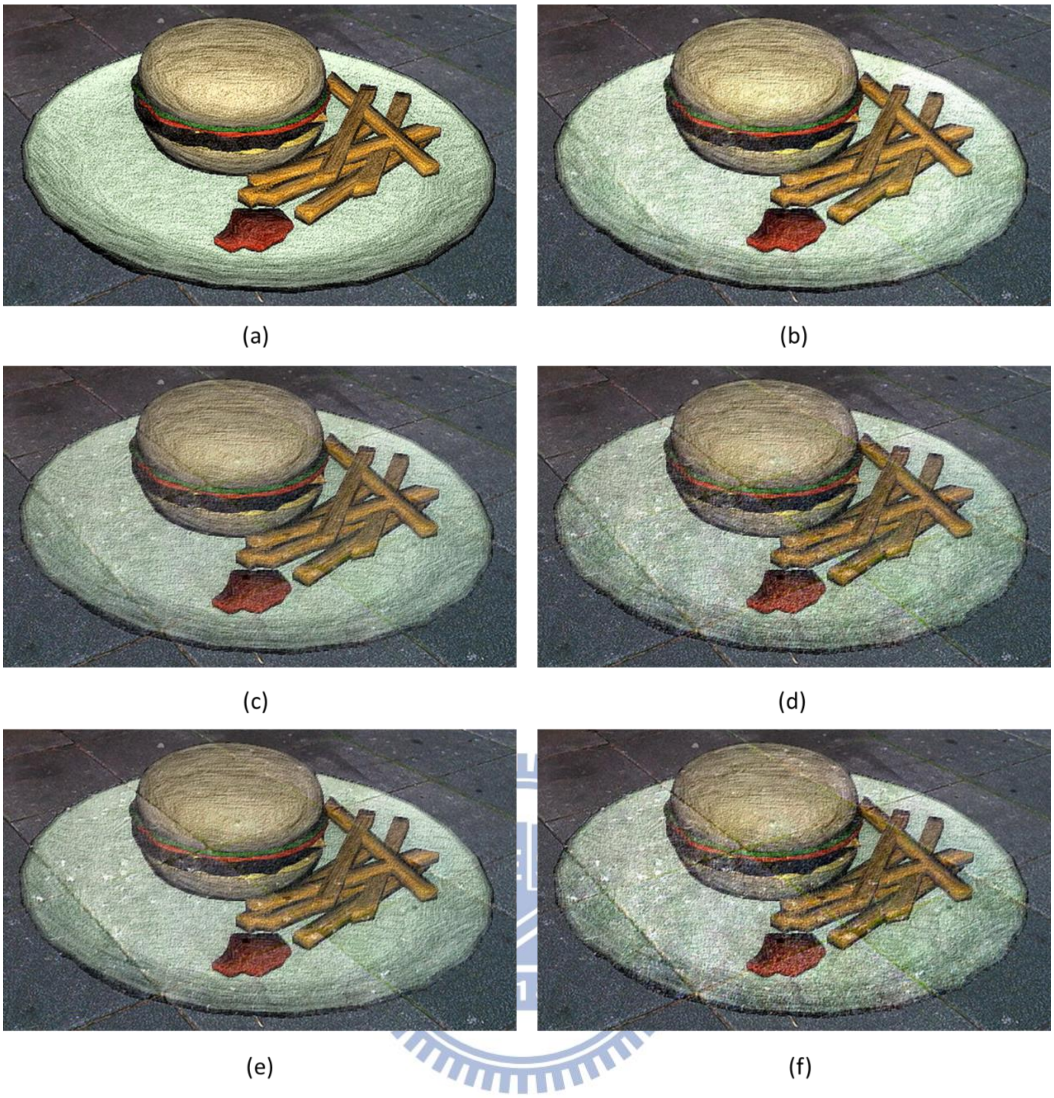
Figure 3.8: Final blending result.

Chapter 4
Implementation and Results

In this chapter, we present some implementation results. All the synthetic objects are available from Google 3D Warehouse, as shown in Figure 4.2. The results were produced using an Intel Core i7 930 PC with 2.8GHz CPU and 8GB memory. Even a skilled street artist may spend several days drawing a 3D painting. By using our rendering system, each result would be generated only in few minutes.

Figure 4.1 compared six experimental results with different values of gradient control parameters ρ, α and β respectively. The corresponding values of each result were defined in Table 4.1. Figure 4.1(a) was produced by using copy-and-paste only. A composition artifact occurred since the visual difference between two images is too large. Figure 4.1(b) showed up those pixels which were determined as the details of source image. The edges between tiles appeared slightly. Since not all the street artists would use a large amount of pigments to hide the ground surface completely. Decreasing the value of ρ could intensify the proportion of ground gradients (Figure 4.1(c)-(f)). As α and β both increased, the final result would be depicted by more ground textures.

After observing a great quantity of street paintings drawn by artists, these artworks were classified into two main types here. The first type is that objects show themselves above the ground plane to give viewer a 3D illusion effect. Some example



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.1: Blending results with different values of Poisson parameters. (a) copy-and-paste only, (b)-(f) Poisson blending approach.

Table 4.1: Gradient control parameters - ρ, α and β.

Figure	ρ	α	β
Figure 4.1(a)	none	none	none
Figure 4.1(b)	1.0	1.0	1.0
Figure 4.1(c)	0.7	1.0	1.0
Figure 4.1(d)	0.7	1.0	2.0
Figure 4.1(e)	0.7	2.0	1.0
Figure 4.1(f)	0.7	2.0	2.3

results are shown in Figure 4.3-4.9. The ground gradients of source image in Figure 4.4 and Figure 4.6 were too high, it can barely see the object image. Thus the value of Z could be lowered to restrict the probability of choosing ground gradients as v_{pq}. Conversely, the second type means objects appear to fall into the ground. This effect usually makes people believe there is a new space created on the other side of the plane. In Figure 4.10, we set the object model upside down and unified the line strokes into vertical direction. It looked like a human's reflection was under the ground. Figure 4.11 conveyed that a man was trying to go down the stairs to another place.

Figure 4.12 and Figure 4.13 showed two examples that the synthetic objects could be also rendered on the wall. Sometimes, street paintings would be designed for interacting with people by artists. We imitated such effect in Figure 4.14. A man attempted to pass the drink to the worker. As a whole, our system could synthesize the street photos with any CG model to generate lively 3D street art illusion.

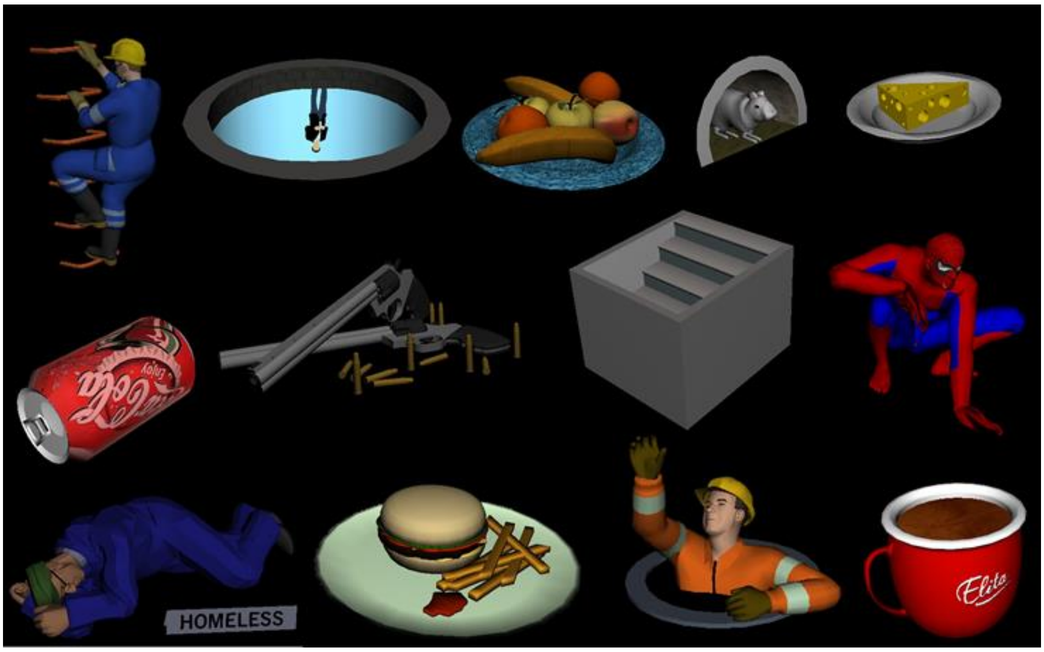


Figure 4.2: Synthetic object models from Google 3D warehouse.

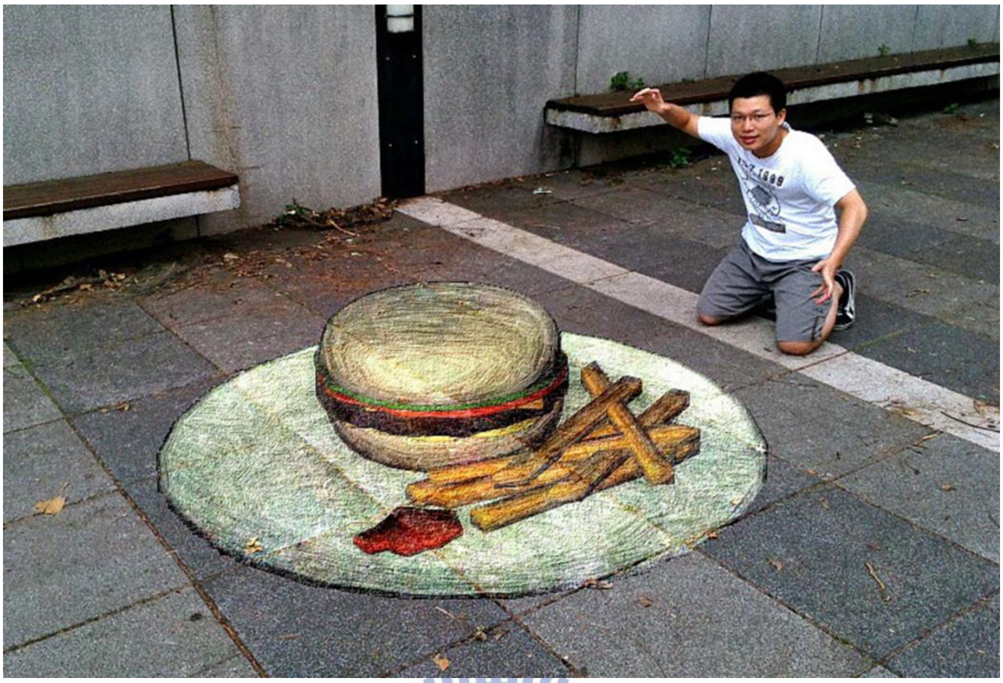


Figure 4.3: The rendered result with fast food model

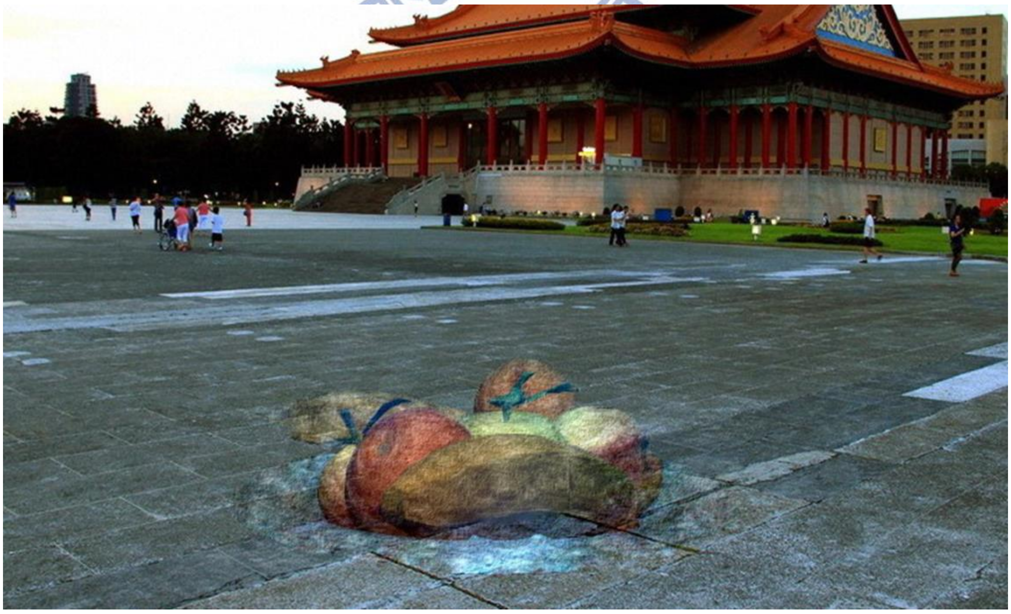


Figure 4.4: The rendered result with fruits model

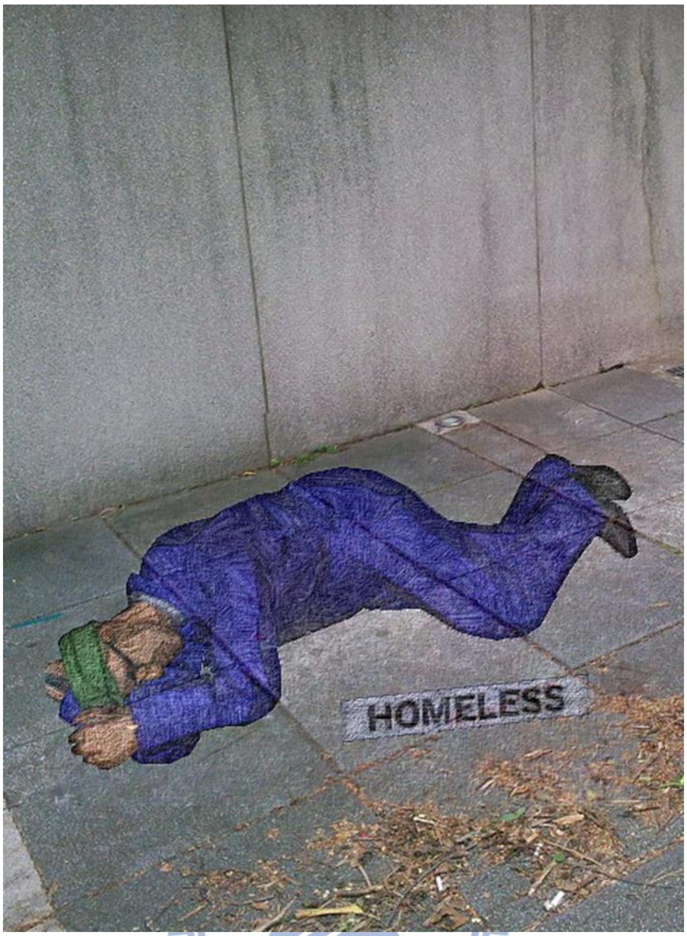


Figure 4.5: The rendered result with human model

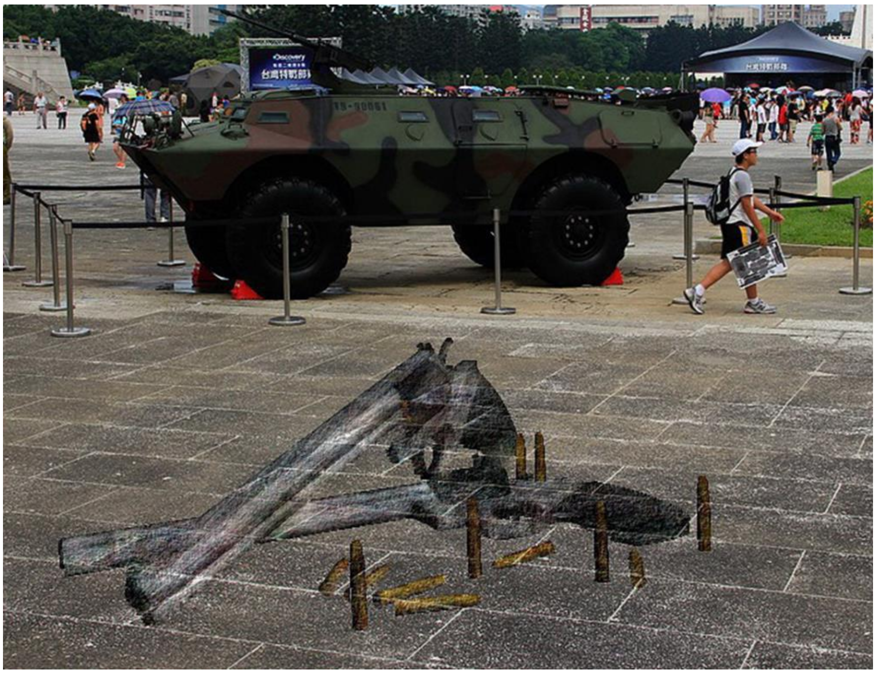


Figure 4.6: The rendered result with revolvers model

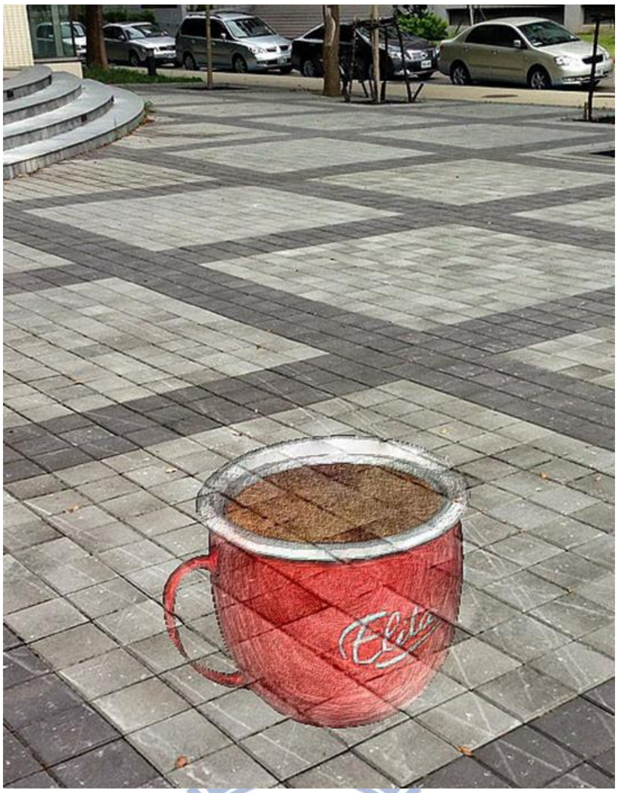


Figure 4.7: The rendered result with coffee cup model

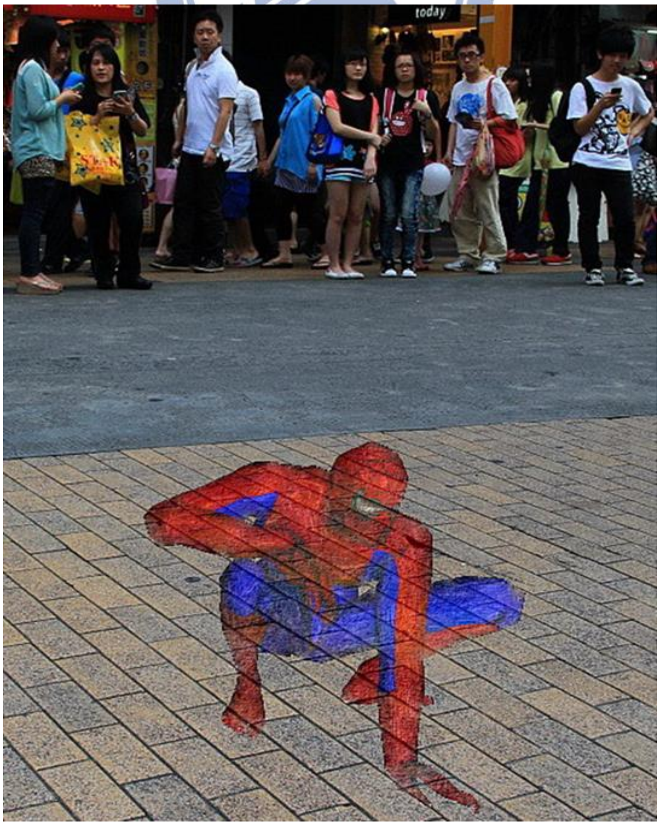


Figure 4.8: The rendered result with spider man model

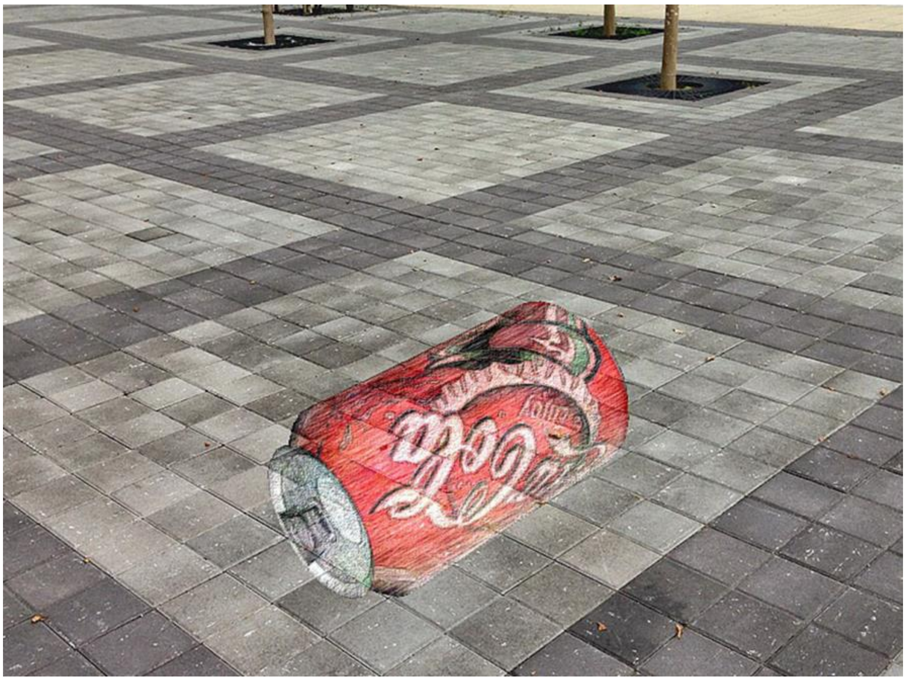


Figure 4.9: The rendered result with Coke can model

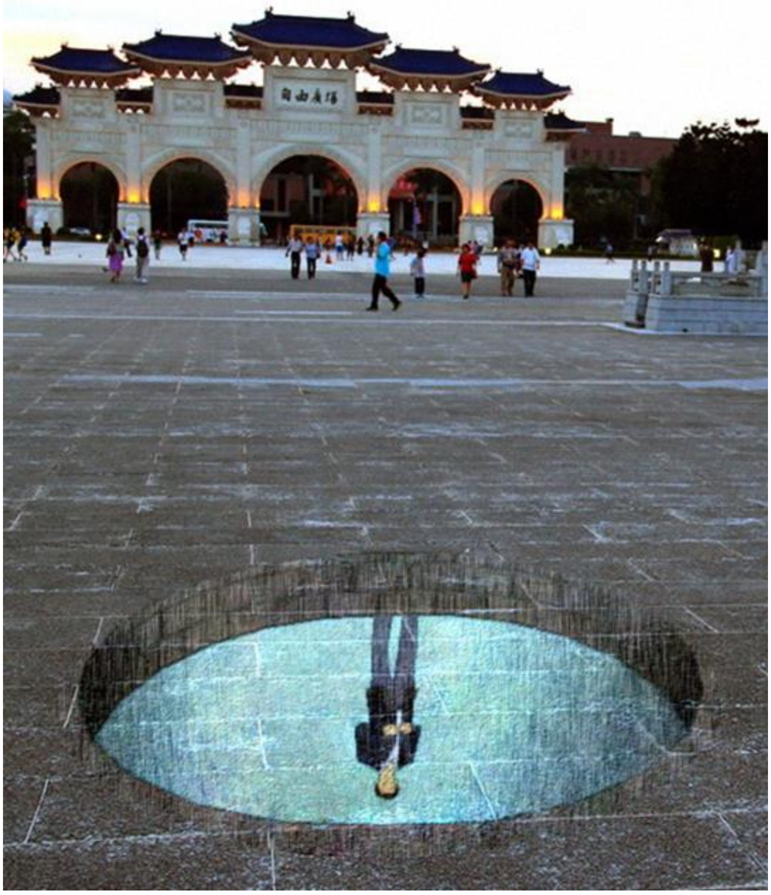


Figure 4.10: Synthetic object appears to fall into ground.

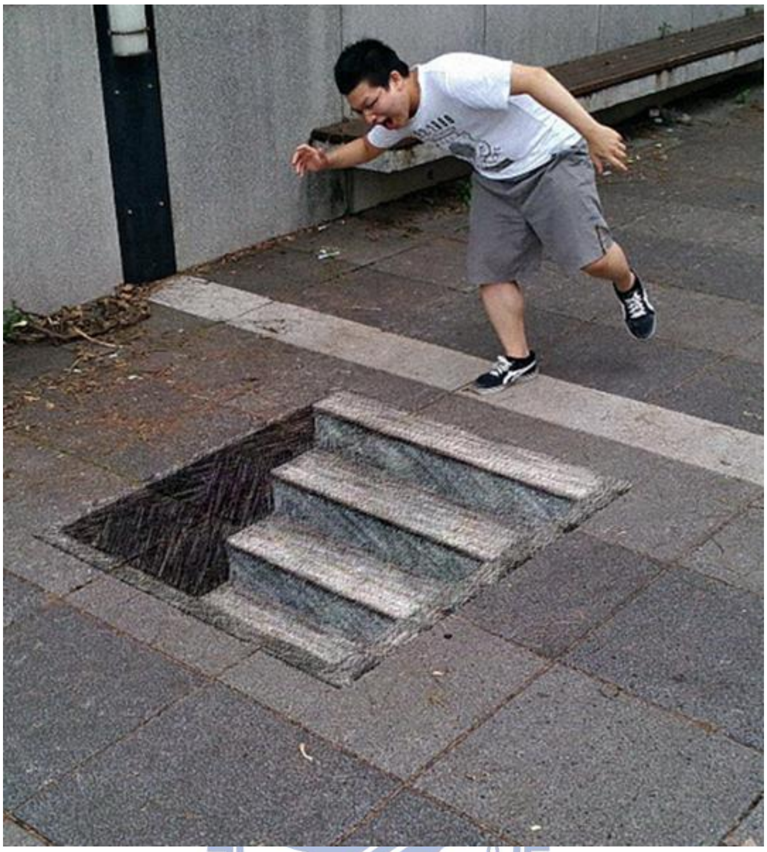


Figure 4.11: Stair model appears to fall into ground.

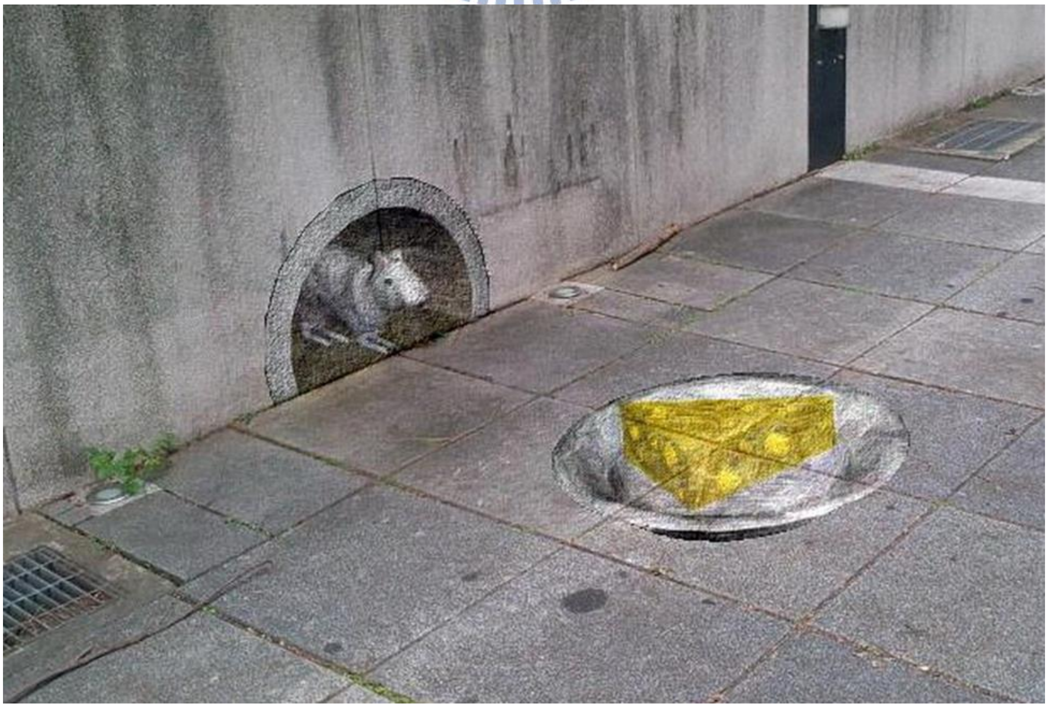


Figure 4.12: The rendered result with rat and cheese cake model

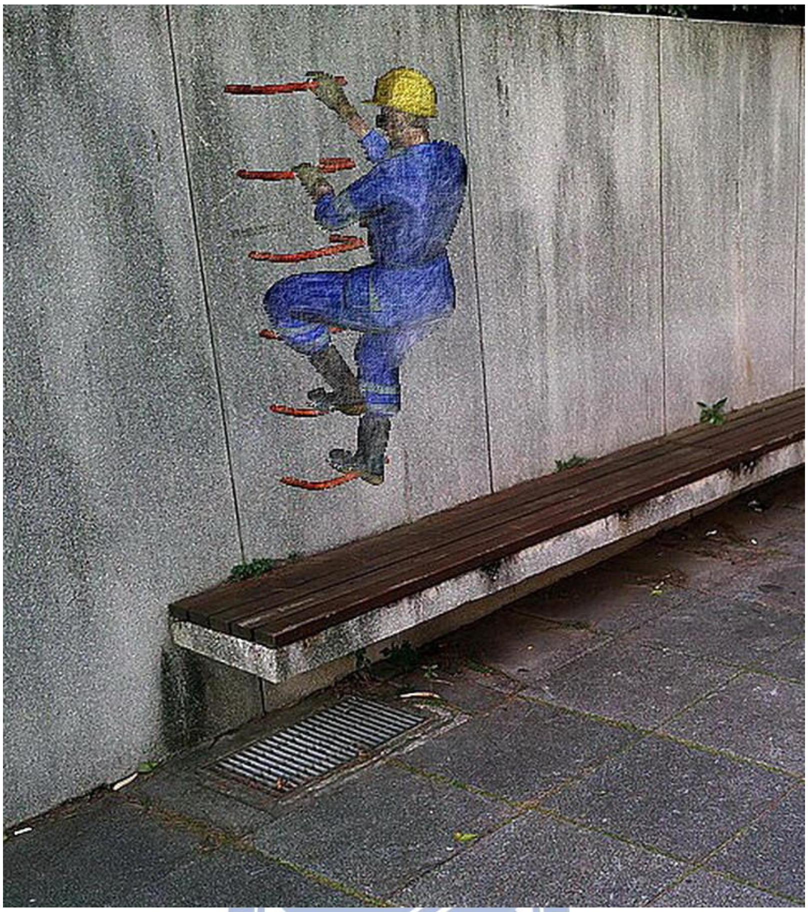


Figure 4.13: The rendered result drawn on wall with worker model

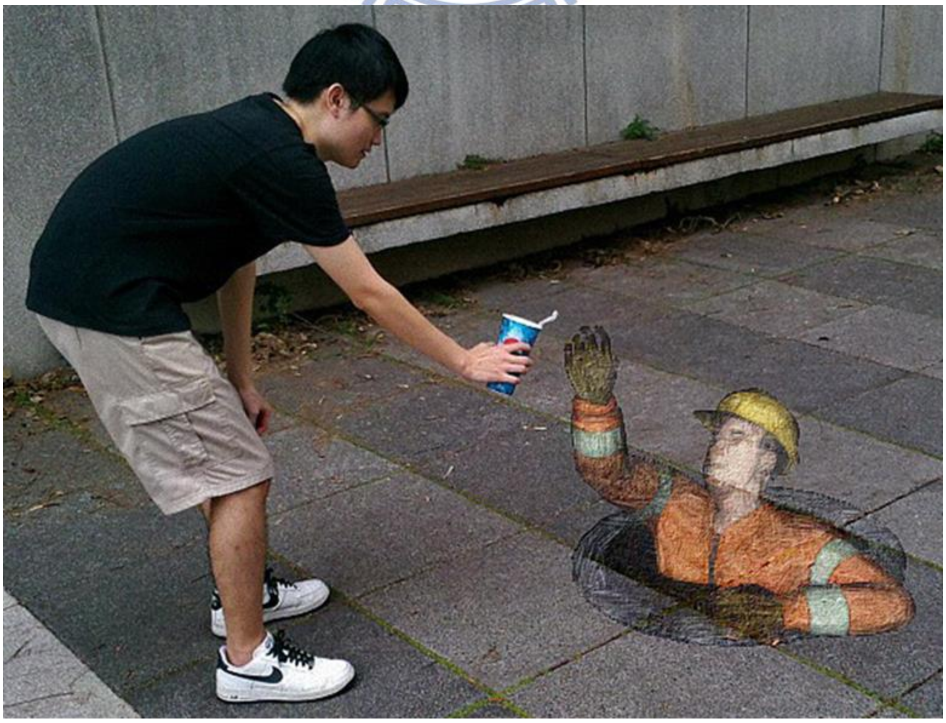


Figure 4.14: The rendered result with worker model

Chapter 5
Conclusion and Future Works

In this thesis, we propose an editing system for generating interesting images which are similar to 3D street art illusion. It synthesizes object models into a realistic photograph without requiring access to the scene to draw 3D street paintings on the ground. The original color presentation in computer graphic is not suitable to convey artist’s painting directly. Thus a non-photorealistic rendering technique is applied to the synthetic object for adding stylization. In order to imitate real street paintings, the ground textures should be taken into consideration. We use a modified Poisson blending approach to determine the guidance field. Those important texture details could be seen above the object image. One limitation of our approach is that if the source image doesn’t have noticeable ground textures, it would fail to produce a reasonable result since the object image might hide the street surface completely.

There are some directions for further research. First, an automatic method which could transfer the material style or mood from a guide image onto the synthetic object would be required. When artists design their own street paintings, they might choose some particular pigments for the artworks to maintain the color consistency between background and synthetic object. We believe other existing NPR techniques could be also applied in the rendering process. The challenge would be how to preserve the stylized effects without being spoiled by the street textures. Finally, decreasing some user interactions is another prior goal in our system. All the synthetic objects compositions and arrangements are done by users. We wish these specifications could be accomplished automatically after users decide what kind of 3D illusion they want to see. Hence a database which can separate object models into different categories previously is needed.

References

[1] Abad F., Camahort E., and Vivó R., Integrating synthetic objects into real scenes. ;In Proceedings of Computers & Graphics. 2003, 5-17.

[2] Caprile B. and Torre V.. Using vanishing points for camera calibration. ; In Proceedings of International Journal of Computer Vision. 1990, 127-139.

[3] Cipolla R., Drummond T., and Robertson D.P., Camera Calibration from Vanishing Points in Image of Architectural Scenes. ;In Proceedings of BMVC. 1999.

[4] Cao X. and Shah M., Camera Calibration and Light Source Estimation from Images with Shadows. ;In Proceedings of CVPR (2). 2005, 918-923.

[5] Cabral B. and Leedom L.C., Imaging vector fields using line integral convolution. ;In Proceedings of SIGGRAPH. 1993, 263-270.

[6] Debevec P.E., Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-based Graphics with Global Illumination and High Dynamic Range Photography. ;In Proceedings of SIGGRAPH. 1998, 189-198.

[7] Fischer J., D Bartz., and Straßer W., Stylized Augmented Reality for Improved Immersion. ;In Proceedings of IEEE Virtual Reality. 2005,195–202.

[8] Guillou E., Meneveaux D., Maisel E., and Bouatouch K.. Using vanishing points for camera calibration and coarse 3D reconstruction from a single image. In Proceedings of The Visual Computer. 2000, 396-410.

[9] Hays J. and Essa I.A., Image and video based painterly animation. ;In Proceedings of NPAR. 2004, 113-120.

[10] Hata M., Toyoura,M. and Mao X., Automatic generation of accentuated pencil drawing with saliency map and LIC. ;In Proceedings of The Visual Computer. 2012, 657-668.

[11] Hertzmann A., A Survey of Stroke-Based Rendering. ;In Proceedings of IEEE Computer Graphics and Applications. 2003, 70-81.

[12] Hertzmann A., Painterly Rendering with Curved Brush Strokes of Multiple Sizes. ;In Proceedings of SIGGRAPH. 1998, 453-460.

[13] Hertzmann A., Fast paint texture. ;In Proceedings of NPAR. 2002, 91-91.

[14] Karsch K., Hedau V., Forsyth D.A., and Hoiem D., Rendering synthetic objects into legacy photographs. ;In Proceedings of ACM Trans. Graph.. 2011, 157-157.

[15] Lopez-Moreno J., Hadap S., Reinhard E., and Gutierrez D., Compositing images through light source detection. ;In Proceedings of Computers & Graphics. 2010, 698-707.

[16] Litwinowicz P., Processing images and video for an impressionist effect. ;In Proceedings of SIGGRAPH. 1997, 407-414.

[17] Lu J., Sander,P.V. and Finkelstein A., Interactive painterly stylization of images, videos and 3D animations. ;In Proceedings of SI3D. 2010, 127-134.

[18] Lu C., Xu L., and Jia J., Combining sketch and tone for pencil drawing production. ;In Proceedings of Expressive. 2012, 65-73.

[19] Li N. and Huang Z., A feature-based pencil drawing method. ;In Proceedings of GRAPHITE. 2003, 135-140.

[20] Mao X., Kikukawa M., Kashio K., and Imamiya A., Automatic Generation of Pencil Drawing Using LIC. ;In Proceedings of SIGGRAPH. 2002, 149-149.

[21] Olsen S.C., Maxwell B.A., and Gooch B., Interactive vector fields for painterly rendering. ;In Proceedings of Graphics Interface. 2005, 241-247.

[22] O'Donovan P. and Hertzmann A., AniPaint: Interactive Painterly Animation from Video. ;In Proceedings of IEEE Trans. Vis. Comput. Graph.. 2012, 475-487.

[23] Pérez P., Gangnet M., and Blake A., Poisson image editing. In Proceedings of ACM Trans. Graph.. 2003, 313-318.

[24] Sousa M.C. and Buchanan J.W., Observational Model of Blenders and Erasers in Computer-Generated Pencil Rendering. ;In Proceedings of Graphics Interface. 1999, 157-166.

[25] Tsai R.. A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology using Off-the-shell TV Cameras and Lenses. ; In Proceedings of IEEE Journal. Robotics and Automation, Vol. 3. 1987, 323-344.

[26] Wang B., Wang W., Yang H., and Sun J., Efficient Example-Based Painting and Synthesis of 2D Directional Texture. ;In Proceedings of IEEE Trans. Vis. Comput. Graph.. 2004, 266-277.

[27] Yamamoto S., Mao X., and Imamiya A., Enhanced LIC Pencil Filter. ; In Proceedings of CGIV. 2004, 251-256.

[28] Yamamoto S., Mao X., and Imamiya A., Colored Pencil Filter with Custom Colors. ;In Proceedings of Pacific Conference on Computer Graphics and Applications. 2004, 329-338.

[29] Zeng K., Zhao M., Xiong C., and Zhu S.C., From image parsing to pai

nterly rendering. ;In Proceedings of ACM Trans. Graph.. 2009.

[30] Zhang Z., A Flexible New Technique for Camera Calibration. ;In Proceedings of IEEE Trans. Pattern Anal. Mach. Intell.. 2000, 1330-1334.

[31] Zhao M. and S.C. Zhu, Customizing painterly rendering styles using stroke processes. ;In Proceedings of NPAR. 2011, 137-146.

[32] Fringe Festival. Live Performance 3D Street Painting in Amsterdam, the Netherlands. 2012. http://wetalkchalk.com/portfolio/3d-street-painting-fringe-festival-amsterdam/

[33] Lelystart - The Netherlands. Hollands first 3D street painting festival in Lelystad. 2011. http://wetalkchalk.com/portfolio/3d-street-painting-princess-and-the-pea/