Efficient view manipulation for cuboid-structured images

Yanwen Guo a,*, Guiping Zhang a, Zili Lan a, Wenping Wang b

a State Key Lab for Novel Software Technology, Nanjing University, PR China
b Department of Computer Science, The University of Hong Kong, Hong Kong

1. Introduction

Advances in imaging technology and hardware improvement of digital cameras result in continuous improvements of image quality. People can take high quality photos at high resolutions without always suffering from noises, low contrast, and blur that may degrade photo quality, more easily than before. However, photos taken by amateur photographers are often with bad viewpoints, for instance slanted man-made structures and unbalanced compositions, making scenes look dull and less vivid. On the other hand, when looking at a photo shared by friends or downloaded from Flicker or Photobucket, people may imagine naturally what the scene looks like looking at a photo shared by friends or downloaded from Flicker or Photobucket. Such images are very popular, for example, the photos of buildings (upper left and indoor scenes (upper right), apartments, and buses. Essentially, some photos exhibit latent cuboid structures. For example, the lower right photo of Fig. 1 is such an image since we can easily construct two perpendicular planes by specifying auxiliary lines shown as the dotted lines in this photo, even though physically one of the two planes containing the auxiliary lines does not exist. Such a cuboid structure is the major visual cue to depict a three-dimensional scene and to convey perspective. By manipulating the cuboid structure reconstructed by acceptable

Stereoscopic devices and content relying on stereopsis are now widely available, and the problem of manipulating perspective in stereoscopic pairs is addressed in [2]. Assuming that depth variations of the scene relative to its distance from the camera are small, slanted man-made structures can be straightened up by an improved homography model [3].

We do not intend to study the aesthetics of whether or not a photograph looks visually pleasing under the current viewpoint. Instead, our goal is to enable the generation of novel images with new viewpoints given only a single image as input, with moderate user assistance. To this end, our primary observation is that many images of man-made scenes exhibit the cuboid dominated three-dimensional structures, in which projections of two perpendicular planes dominating the latent three-dimensional geometry, occupying the major part of an image. A pair of projected parallel lines in each plane can be found in the image. Such an image either itself has a cuboid structure or its scene is dominated by a cuboid-like object. As shown in Fig. 1, the cuboid-structured images are very popular, for example, the photos of buildings (upper left and lower left), indoor scenes (upper right), apartments, and buses. Essentially, some photos exhibit latent cuboid structures. For example, the lower right photo of Fig. 1 is such an image since we can easily construct two perpendicular planes by specifying auxiliary lines shown as the dotted lines in this photo, even though physically one of the two planes containing the auxiliary lines does not exist. Such a cuboid structure is the major visual cue to depict a three-dimensional scene and to convey perspective. By manipulating the cuboid structure reconstructed by acceptable
A spidery mesh is employed to obtain a simple scene model from the central perspective image using a graphical interface. The animators utilize this incomplete scene information to make animation from the input pictures. Instead of attempting to recover precise geometry, a rough 3D environment is constructed from a single image by applying a statistical framework [8]. The model is constructed directly from the learned geometric labels: ground, vertical, and sky, on the image. None of the above methods aim to re-generate a new image with high visual quality as if it is captured from a novel viewpoint. In contrast, we only need to partially recover a cuboid dominated 3D representation of the image with moderate user interaction, the whole image is re-rendered by making the rest image region deform in accordance with the re-projection of the cuboid structure.

Recently, the advances of shape deformation [9,10] and retargeting techniques [11–15] make it possible to manipulate perspective by the means of image deformation [1]. A 2D image warp is computed by optimizing an energy function such that the entire warp is as shape-preserving as possible, and meanwhile satisfies the constraints originated from projective geometry. The user first annotates an image by marking a number of image space constraints, with pixel accuracy. User assistance is required to accurately mark the image and manipulate its perspective with a number of image space constraints. Overall eight different types of constraints which may oppose directly each other are incorporated into the energy function. Taking care of these constraints cautiously for efficient optimization poses a challenge for amateur users. The problem of manipulating perspective in stereoscopic pairs is addressed in [2]. Given a new perspective, correspondence constraints between stereoscopic image pairs are determined, and a warp for each image which preserves salient image features and guarantees proper stereopsis relative to the new camera is computed. Perspective projections are limited to fairly narrow view angles. Correction of image deformations incurred by projecting wide fields of view onto a flat 2D display surface is address in [16,17].

Our work is also inspired by the recent efforts on photo composition assessment and enhancement [18–20]. Most methods build their measures of visual aesthetics on the rule of thirds which means that an image should be imaged as divided into nine equal parts by two equally spaced horizontal lines and two vertical lines, and important compositional elements should be placed along these lines or their intersections. Bhattacharya et al. [18] learn a support vector regression model for capturing aesthetics. Image quality is improved by recomposing the salient object onto the inpainted background or by using a visual weight balancing technique. Liu et al. [20] modify image composition by using a compound operator of crop-and-retarget and seek the solution by particle swarm optimization.

3. View manipulation of the cuboid structure

Our view manipulation method is specifically designed to optimize viewpoints of those images that show cuboid-dominated three-dimensional structures. Extracting a 3D representation from a single-view image depicting a 3D object has been a longstanding goal of computer vision. It has been shown recently that 3D cuboids in single-view images can be automatically localized by using a discriminative parts-based detector [21]. We allow the users to interactively specify projected lines of the latent cuboid structure on the image, with which we estimate an approximation of the cuboid geometry. Hough transform and Canny edge detector are used to assist users and to reduce interaction errors in this process. We show that, given a new viewpoint, the re-projection of this
approximated cuboid structure is sufficient to meet the requirement of accuracy of viewpoint change.

3.1. A standard case

3.1.1. Cuboid reconstruction

For ease of exposition, it is initially assumed that we can find, on the image plane, the projections of a vertical edge and two pairs of parallel edges with two interaction points on the vertical edge of a standard cuboid structure, as shown in Fig. 2. Without loss of generality, we assume that the imaging plane is placed along with the XY plane of the world coordinate system with its center at the world origin. The camera is stationed at Z-axis with center of projection $O(0,0,f)$. $f$ is the focal length of the camera. Let $P_0$ and $P_1$ are two corners of the cuboid shared by two perpendicular planes $P_1(p_1, p_2, p_3)$ and $P_2(p_0, P_1, P_2, P_3)$ on the three-dimensional cuboid. Since $P_0 P_1$ and $P_0 P_2$ meet at a vanishing point $c_1$, except for the special case $P_0 P_2 / P_0 P_2$. Similarly, the extensions of $p_1 p_3$ and $p_2 p_3$ meet at another one $c_2$. Imagine that $P_0 P_1$ and $P_0 P_2$ meet at a point at infinity whose projection on the imaging plane is $c_1$. The extension of $F_1$ will meet with $P_0 a$ and $P_0 a$, $P_0 a$ at this point as well. We thus have $F_1 / P_0 P_2$ and $F_1 / P_0 P_2$, and $F_1 / P_1$. Similarly, $F_2$ is parallel to $P_2$, and $F_3$ is perpendicular to $F_2$ with which $f$ can be obtained easily. It is still impossible to recover the accurate geometry and position of the 3D cuboid structure without any other prior knowledge. By making reasonable assumptions, we wish to generate an approximation of the structure which is exactly the same as the accurate one without considering the scale difference. Given any new viewpoint, we show that the re-projection of this approximated cuboid structure is sufficient to meet the requirement of accuracy of viewpoint changes.

Considering that $||P_0 P_1||$ is proportional to $||P_0 P_1||$, coordinate of $P_0$ can be obtained by setting the ratio of $||P_0 P_1||$ to $||P_0 P_1||$ to a constant. We set it to 1 in our experiments. $P_1$ can be represented by parametric coordinate with $F$ and $p_0$. We then compute $P_1, ..., P_5$ by exploiting the geometric relationships

$$\begin{align*}
    &P_0 P_1 = |P_0 P_1|, P_0 P_4 = |P_0 P_4|, P_0 P_2 = |P_0 P_2|, P_0 P_3 = |P_0 P_3|, P_0 P_5 = |P_0 P_5|.
\end{align*}$$

3.1.2. Analysis of accuracy

Given a new viewpoint, we assume that focal length $f$ of the camera remains fixed, since zooming in and out can be easily imitated by upsampling and downsampling the image. Without loss of generality, center of the scene is placed at $(0, 0, z_0)$ with $z_0$ the z-coordinate of $P_0$. Recall that we set the ratio of $||P_0 P||$ to $||P_0||$ to 1. Therefore, center of the scene is $(0, 0, 0)$ which is in accordance with the world center $O$. Let us denote the new viewpoint by $O(\phi \sin \varphi \cos \theta, f \sin \varphi \sin \theta, f \cos \varphi)$ where $(\theta, \varphi)$ is the polar angle and azimuthal angle in spherical coordinate system. The new imaging plane passing through the world center $O$ whose normal vector is $\overrightarrow{O}$. Let $(Q_0, ..., Q_5)$ denote the real coordinates of the 3D cuboid, corresponding to the projected points $(p_0, ..., p_1)$ on the input image. We prove that the 2D projections of $(P_0, ..., P_1)$ on the new imaging plane $I$ is identical to the projections of $(Q_0, ..., Q_5)$, by eliminating translation. To achieve this, we only need to prove

$$\overrightarrow{P_0 P_1} = q_0 \overrightarrow{Q_0 Q_1},$$

where $q_0 P_1$ represents the projected vector of $P_0 P_1$ on $I$, and $q_0 Q_1$ denotes the projected vector of $Q_0 Q_1$ on $I$, the imaging plane with focal length $f$ if $(Q_0, ..., Q_5)$ are given (see Fig. 3).

We should rotate the camera around $O(0, 0, z_0)$ if $(Q_0, ..., Q_5)$ are known. In this case, the new viewpoint is $O$. We have $|Q_0 F| = |O Q_0|$. Let $I$ passing through $O$ be an auxiliary plane whose normal is $Q_0 O$. Let $P_0 Q_1$ be the projected vector of $Q_0 Q_1$ on $I$. We get

$$q_0 Q_1 = f |O Q_0 F| P_0 Q_1.$$

It is obvious that $\Delta(O Q_0 F)$ and $\Delta(O Q_0 P_0 P_1)$ are similar triangles. $\Delta(P_0 F_1 O P_1)$ and $\Delta(P_0 O Q_0 P_0 P_1)$ are similar triangles. We get

$$\frac{P_0 Q_1}{P_0 P_1} = \frac{O Q_1}{O Q_0} = \frac{O Q_1}{|O Q_1|} = \frac{f |O Q_0 F|}{P_0 Q_1}.$$

Combining (3) and (4), we get $P_0 P_1 = q_0 Q_1$.

3.2. A more complex case

In some images we cannot find, on the projected vertical edge, two interaction points of the projected parallel edges. Fig. 4 shows such an example. To tackle this issue, we first compute the vanishing points of the projected parallel line segments. Two auxiliary lines, shown as dotted segments in Fig. 4, which pass through the corresponding vanishing points can be drawn. This case is then converted to the standard case we have described previously. We then compute equations of the two perpendicular

Fig. 2. Left: a standard case of the projection of a cuboid structure. Note that, $|P_0 P_1| = |P_0 P_1|$ and $|P_0 P_1| = |P_0 P_1|$ are not required. $P_0 P_1$ and $P_0 P_1$ are not necessarily the projections of two vertical edges of the cuboid. Right: the 3D cuboid structure.

Fig. 3. Projection of $P_0 P_1$ on the new imaging plane $I$ is identical to the projected $Q_0 Q_1$ on $I$, by eliminating translation.
planes with which spatial coordinates of the eight endpoints of the originally specified line segments can be easily obtained.

4. Cuboid-guided image warp

Given a new viewpoint, the cuboid structure is projected onto the new imaging plane. The new image is rendered by making the rest image region deform in accordance with the transformation of the cuboid structure. We use a mesh representation to realize image deformation as shown in Fig. 5. Generating a mesh for an image for the tasks of image resizing and manipulation has been discussed in [13,14,1,22,23]. Unlike the quad mesh employed by most previous methods, we use triangular mesh to represent the input image. An advantage of the triangular mesh over quad mesh is that cuboid-structured region-of-interest (ROI) can be represented compactly by the meshes with moderate density, since it is possible to approximate the structures by triangle edges, and they can be easily obtained.

We use constrained Delaunay triangulation to create a content-aware mesh representation. Points are first evenly sampled from image borders, the cuboid structure, and strong edges detected using Hough transform, and their connectivity are constraints for triangulation. To keep uniformity of point density, we detect some corners and if necessary further add some auxiliary ones, since nearly uniform point density normally facilitates mesh processing.

We represent the triangular mesh as \( \mathbf{M} = (\mathbf{V}, \mathbf{E}, \mathbf{T}) \) with vertices \( \mathbf{V} \), edges \( \mathbf{E} \), and triangles \( \mathbf{T} \). We use \( \mathbf{V} = [v_0, v_1, \ldots, v_n] \) with \( v_i = (x_i, y_i) \in \mathbb{R}^2 \) to denote initial vertex positions.

For a new viewpoint, the cuboid structure is re-projected using the approximated geometry. With this re-projected cuboid structure, we render the new image with high visual quality by making the approximated geometry. With this re-projected cuboid structure and other important visual features.

\[ E \mathbf{T}_{1j} = a_1 x_j + a_2 y_j + a_3 x_k + b_1 y_j + b_2 y_k + b_3 y_k = 0. \]

We define the total conformality energy by summing up the individual energy terms on each triangle.

\[ E_s = \sum E^2_{\mathbf{T}_{1j}}. \]
**Line constraint:** Strong edges such as straight lines are important visual features. They are vital clues for understanding image content, and should be maintained as-rigid-as possible. We detect those line segments using Hough transform. Users can also specify some additional curved edges. Points sampled from the edges and their connectivity yielded from the corresponding edges are fed into the triangulation process beforehand. Let \((\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k)\) denote a triplet of vertices on a straight line. To preserve the shape of strong edges, we preserve the length ratio \(l_i\) of \(\mathbf{v}_i\) to \(\mathbf{v}_j\), and the angle \(\theta_i\) formed by \(\mathbf{v}_i\) and \(\mathbf{v}_j\) in each triplet locally [15]. We express the energy term regarding line constraint as

\[
E_L = \sum_{(\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k)} \left\| (\mathbf{v}_k - \mathbf{v}_j) - R_j \cdot (\mathbf{v}_j - \mathbf{v}_i) \right\|^2, 
\]

with

\[
R_j = \begin{pmatrix} \cos \theta_j & \sin \theta_j \\ -\sin \theta_j & \cos \theta_j \end{pmatrix}.
\]

Besides the lines detected by Hough transform or specified by the user, we use line constraint to preserve the shapes of those salient objects that lie across two different faces of the latent cuboid or the cuboid and the rest image region. A line segment is constrained similarly.

**Border constraint:** Physically each side of image borders should be constrained to remain straight. The energy term \(E_B\) of this constraint is defined similarly to the line constraint.

**Total energy:** In summary, combining all the above energy terms, we wish to minimize the following energy function:

\[
\arg \max \sum \lambda_s E_s + \sum \lambda_c E_c + \sum \lambda_v E_v + \lambda_B E_B,
\]

subject to \(F_c(\mathbf{v}_1, \ldots, \mathbf{v}_n) = 0\). (14)

where \(\lambda_s\), \(\lambda_c\), \(\lambda_v\), and \(\lambda_B\) are the coefficients weighting different energy terms. Straight and important curved lines as visually prominent features should be kept. To mimic hard constraints, \(\lambda_c\), the weight of straight line constraint, is often set to a bigger value compared with the weight of shape constraint, \(\lambda_v\), the weight of vertical and horizontal line constraint, can be set by the user with respect to image content. Obviously, \(E_B\) can be enforced as a hard constraint with a bigger \(\lambda_B\). It is useful to straighten up those slanted man-made structures in an input image to improve its perceptual quality. In practice, to deal with possible confliction between border constraint and constraint on the cuboid structure near image border, \(E_B\) often takes effect as a soft constraint by setting \(\lambda_B\) to a small value. For all the results in this paper we use weights of \(\lambda_s = 1\), \(\lambda_c = 100\), \(\lambda_v\), and \(\lambda_B\) are set to 100 as well if \(E_B\) and \(E_v\) are taken as hard constraints, and are set to 10 for soft constraints.

In is noted that we do not impose the constraint for avoiding mesh flip-over in the above energy function. In all our experiments, mesh flipping is seldom encountered. We check it after the deformed mesh is obtained, and once flipping is detected, we correct them locally.

The energy function is a quadratic function of \(\mathbf{v}\). The solution can be obtained efficiently by solving a sparse linear system.

5. **Experiments**

We have implemented our view manipulation algorithm on a PC with Intel Core i3-2100 CPU at 3.1 GHz, and experimented with our technique on a variety of images. Some representative results are shown in Figs. 6–10 and 12.

Figs. 6 and 7 demonstrate the results on several images of man-made buildings. The first and third rows are the input images.
The standard cuboid structures of the interest objects, shown as blue line segments specified by the user on the images, can be found in these images. We specify them manually in our current implementation. Note that, for each pair of parallel edges of the cuboid structure, lengths of the edges that are parallel to each other are not necessarily identical, as shown by these examples. The second and fourth rows are the corresponding results with changed viewpoints. The results of viewpoint shift on two indoor images are shown in Fig. 8.

In Fig. 9 we demonstrate an example where user-specified line segments in blue cannot meet the standard of a cuboid structure. We add auxiliary lines, shown as dotted blue segments, under the constraint of vanishing point, allowing for better control over image deformation. The green segments are detected by using...
Hough transform and they are subject to the line constraint and vanishing point constraint in our optimization. Two new images rendered under new viewpoints are shown.

Fig. 10 shows the results of viewpoint adjustment for a photograph taken in the downtown of Toronto. The center of this photograph is a building with cuboid structure. Many tall buildings and a viaduct in the background make viewpoint adjustment of this photo a challenging task. Two new images with changed viewpoints are given.

5.1. Comparison with previous work

We first compare our results against the results produced by [1] (Fig. 11). Our results are comparable to those by [1]. In [1], the user needs to annotate an image by marking a number of image space constraints which are fed into the optimization framework. The user needs to understand the basic principles of perspective construction, such as vanishing points and lines, in order to use their tool properly. This, to some extent professional work may make image manipulation a frustrating task for users, and may cause failure and even improper perspective due to improper interaction operations. In contrast, to use our framework the user only specifies the cuboid structure, and the knowledge on perspective projection is not needed. Furthermore, we can generate novel images under key viewpoints around the viewpoint of input image, with given viewing angles, due to the reconstructed approximate cuboid structure. This is demonstrated by the application of viewing sphere as follows. By contrast, the algorithm in [1] cannot manipulate the image in an intuitive manner as ours with controlled viewpoint. Table 1 shows the number of line segments for constraints used in each image. We use six line segments which include five on the cuboid structure and one for the extra line constraint for the middle image shown in Fig. 6. For the input image of Fig. 10, 18 line segments are used. Normally, around 15 lines segments are used, the number varying according to image complexity. It should be noted that we do no need to impose line constraint for each straight line in the image, and evenly distributed sparse line segments are enough for constraining the resulting image from obvious line distortions. By comparison, it is reported by [1] that a total of 22 line segments and polygon edges are specified per image by using their method.

Table 1

<table>
<thead>
<tr>
<th>Fig. 6 (left/middle/right)</th>
<th>Fig. 7 (left/middle/right)</th>
<th>Fig. 8 (left/right)</th>
<th>Fig. 9</th>
<th>Fig. 10</th>
<th>Fig. 11 (up/bottom)</th>
</tr>
</thead>
<tbody>
<tr>
<td>13/6/17</td>
<td>16/13/16</td>
<td>13/13</td>
<td>12</td>
<td>18</td>
<td>15/17</td>
</tr>
</tbody>
</table>

The algorithm of [1] needs to minimize a nonlinear least squares function. The optimization time is dominated by solving the linear system at each iteration of the optimization. As reported, it takes on average 3.37 second for optimization. In contrast, computation of our approach is mainly consumed by optimizing Eq. (14) which results in solving a sparse linear system only once due to the nature of quadratic formulation of our algorithm. Normally, a source image is divided into a sparse mesh, with several hundreds of vertexes. It takes around 100 ms to generate a resulting image except for user interaction. Obviously, our algorithm is more efficient than [1].

Our approach focuses on manipulating views of cuboid-structured images. A relevant algorithm on image editing by means of cuboid proxies for representing those cuboid-like objects in man-made environments is given in [6]. We would like to clarify the difference between our approach and that of [6]. First, we have different goals. Our approach wishes to change the viewpoints of images with cuboid dominated structures. The algorithm in [6], however, aims at editing the cuboid-like objects mimicking real-world behavior such as replacing the objects with new ones and deforming the objects in images. Viewpoint
changing of the environment is not supported. Second, and more importantly, to meet the different goals, only partial and inaccurate recovery of the standard and non-standard cuboid structures is required by our approach. We do not need to accurately reconstruct the geometry of the cuboid structure. Only five edges on the cuboid dominated structure for an input image, including a so-called vertical edge and two pairs of parallel edges (not necessarily of identical length) are enough. This is validated by most of our experimental results. We believe it is challenging to automatically detect the cuboid structures and reconstruct the accurate geometry for most our testing images, such as most interest objects in Figs. 6 and 7 and the indoor images in Fig. 8, by algorithm in [6]. Furthermore, our approach is competent for viewpoint manipulations for the image with a non-standard cuboid dominated structure like the input photo shown in Fig. 9.

5.2. Upright adjustment

Man-made structures often appear to be slanted in photos taken by casual photographers. An example is shown in the 1st column of Fig. 12. This is partly due to the improper position where the camera is placed at. Human visual system however always expects tall man-made structures to be straight-up. In [3], the slanted structures are dealt with by using an improved homography model. Our algorithm can also be used to straighten up the slanted cuboid structures in images. The idea is to regenerate the image by modifying the viewpoint of the input image properly. The new viewpoint is computed automatically by letting the projected structures to be vertical.

A software implementation of [3] is Adobe Lightroom Upright. We thus compare our results to those produced by Lightroom. Fig. 12 shows the results produced by Lightroom (2nd row) and those by our algorithm (3rd row). Our results are generally comparable to those by Lightroom upright. The results of Lightroom show apparent visual artifacts in the results of Taj Mahal and Church photos (see the red ellipses). In [3], the authors assume that depth variations of the scene relative to its distance from the camera are small. The reason for the artifacts may be that such assumption does not hold exactly for the two images. Transforming the input image with a homography is not always sufficient since it is oblivious to the depth variations of the latent scenes.
Viewing sphere: We can generate novel images under key viewpoints around the viewpoint of input image, with given viewing angles. This enables us to design an interface through which the user can watch the scene by changing viewpoints smoothly on a viewing sphere, mimicking 3D browsing experience. As shown in Fig. 13, the images on the viewing sphere under new viewpoints are interpolated from the original input and the newly rendered images under four key viewpoints. Please refer to our accompanied files for the live demo.

Limitations: Our system creates a partial reconstruction from a single image. Although visually pleasing results are generated by our system for a variety of images with standard or non-standard cuboid structures, its applicability is limited. Furthermore, the new viewpoints are restricted to a certain range around the viewpoint cuboid structures, its applicability is limited. Furthermore, the new viewpoints are restricted to a certain range around the viewpoint.

Fig. 14: A failure case. The roof of the house is not visible in the left input and its corresponding part looks hollow in the right result due to visibility shifting caused by viewpoint change.

6. Conclusion

We have presented an algorithm for manipulating the viewpoints of those cuboid-structured images and generating new images realistically. Our framework creates partial scene reconstruction with minimal user interaction and we show that such an approximate reconstruction is sufficient to re-render the image under a new viewpoint, via a triangular mesh deformation scheme. The mesh deformation energy is optimized efficiently by solving a sparse linear system. In addition to the generation of images with novel viewpoints, we provide a user interface that allows users to watch the scene under new viewpoints on a viewing sphere interactively.

In the current implementation, the user needs to manually specify lines of the latent cuboid structure on the input image, even though Hough transform and Canny detector can be used to assist in this operation. Recent research efforts on localizing 3D cuboids in single-view images may facilitate automation of this process [21,6]. In future we plan to explore the possibility that automatic detection and analysis of the cuboid structures are integrated into our framework.

Acknowledgments

We greatly thank the anonymous reviewers for the positive and constructive comments. This work was supported in part by the National Science Foundation of China under Grants 61073098, 61021062, and 61373059, the National Basic Research Program of China (2010CB327903).

References