On the Use of Ray-tracing for Viewpoint Interpolation in Panoramic Imagery

Feng Shi, Robert Laganière, Eric Dubois
VIVA research lab
SITE, University of Ottawa
Ottawa, ON, Canada, K1N 6N5
{fshi098,laganier,edubois}@site.uottawa.ca

Frédéric Labrosse
Aberystwyth University
Aberystwyth SY23 3DB
United Kingdom
ffl@aber.ac.uk

Abstract

To acquire seamless visualization of environments from different viewing positions and orientations, it is desirable to generate virtual images for an arbitrary position given a set of reference views. In this paper, a simple interpolation method based on ray-tracing is proposed for viewpoint synthesis from panoramas taken with multi-sensor cameras. Instead of attempting to recover a dense 3D reconstruction of the scene, the method estimates the pose between each panorama and then backward project the point along the ray that exhibits the best colour consistency. We show that by limiting the search space using sparse depth information both the speed and the accuracy of the interpolation are improved.

1 Introduction

In the recent years, there has been a great deal of interest in image-based rendering (IBR) techniques for producing novel views from a set of pre-captured reference images. This kind of techniques is often called view synthesis. Compared with traditional geometry-based approaches which render directly from geometric models, the IBR methods can produce more photorealistic results, and avoid tedious, error-prone 3D shape modelling of complex scenes.

View synthesis can be classified into two categories based on viewing freedom: (i) viewing from limited viewpoints; (ii) viewing from arbitrary viewpoints. The first group can only produce virtual images from a restricted set of points of view, providing a limited immersive user experience. In ”View Morphing” [13], the novel views are directly generated by interpolating the corresponding pixels of the input images. Seitz and Dyer adopt a three-step algorithm to guarantee the intermediate view being geometrically correct or 3D shape preserving. To improve view morphing, Lhuillier and Quan [9] propose a quasi-dense matching algorithm based on region growing, and used the joint view triangulation to interpolate novel views. Sun and Dubois [16] improve the quality of triangle-based view morphing by using a feedback-based method to determine the weights to use when combining the textures from different reference images. These methods often limit the new viewpoint to lie on the straight line connecting the projection centers of reference viewpoints. Another typical view synthesis method which also has constrained view point but with a richer user experience is Multi-Center-Of-Projection (MCOP) [12]. MCOP samples the scene by placing the camera around the objects of interest, and interpolates these images to generate virtual view. The navigation is then restricted to a circular trajectory.

The second group of the methods can theoretically generate arbitrary point of view with user-specified rotation and translation. Laveau and Faugeras [8] employ the epipolar constraints to perform a raytracing-like search of corresponding pixels in reference images for novel view pixels. Dense correspondences are computed in their approach. ”Plenoptic Modelling” [10] can allow rendering from arbitrary viewpoints without explicit 3D reconstruction. After cylindrical panoramas are composed, the method computes stereo disparities between cylinder pairs, and then project disparity values to an arbitrary viewpoint. ”Layered Depth Images (LDI)” [15] constructs ”multiple overlapping layers” by using stereo techniques. To render an arbitrary novel view, it is only needed to warp a single image, in which each pixel consists of a list of depth and colour values. These methods require solving the difficult dense or quasi-dense feature correspondence problem to extract disparity or depth values. Dense correspondence is an ill-posed problem, especially when the reference images have large difference in rotation and scale due to viewing orientations and zooming or large baseline separations. Sometimes it is almost impossible to compute dense correspondences due to the untextured and slanted regions in real image.

In this paper, we test an approach to synthesize novel views from panoramas for any specified position and viewing direction that avoids the computation of dense correspondences. Our approach is similar to that of [8] in that we use a raytracing-like algorithm. For every pixel in the new target image, a search is performed to locate the corresponding pixels in reference images. However, instead of computing dense correspondences, we use a colour consistency constraint among multiple panoramas to guide the search. We also show that by limiting the search space using sparse depth information improves both the speed and the accuracy of the interpolation.

This paper is organized as follows: the next section briefly introduces acquisition and geometry of cubic panorama. Section 3 describes the basic ray-tracing interpolation approach and Section 4 proposes two implementations of the algorithm. In Section 5, we present some experimental results. Section 6 is a conclusion.

2 Geometry of Cubic Panoramas

Cubic panoramas have been introduced by Greene [4] and made popular by Apple QuickTime 5 VR. They constitute a convenient representation for $360°$ panoramas as they can be stored and rendered very efficiently in modern graphic hardware [3]. We use here the Point Grey Ladybug camera to capture and generate these panoramas. The Ladybug camera consists of six 1024x768 colour CCDs, with five CCDs positioned in a horizontal ring and one positioned vertically.

After capture, the six raw images, with roughly 80 pixels overlap between neighbouring sensors, need to be stitched together to form a panorama image. This can be done by fusing the six images to form a spherical mesh that is then projected onto six cube faces [**?**]. Figure 1 shows a cubic panoramas where the cube is laid out in a cross pattern, the middle image alignment showing the left, front, right and back faces (since there is no sensor positioned downward, there is a black hole in the bottom face of the cube).

An ideal pinhole camera model is assumed under which a cubic panorama is made of six identical ideal cameras whose optical centers are located at the cube center. Consequently, the calibration matrix **K** and relationships of different faces of cubic panorama are implicitly known given the size of cubic image [5]. Each cube is captured from a point in space, and its location and orientation can be represented with translation vector **t** and rotation matrix **R**. The essential matrix **E** is a compact representation of the relationship between two panoramas. The relative position between two panoramas is therefore obtained as follows:

1. Use Lowe's scale-invariant SIFT [**?**] to detect and match a few, but accurately, features using stringent threshold;

2. Estimate the essential matrix **E** and discard misdetected matches by the robust RANSAC method.

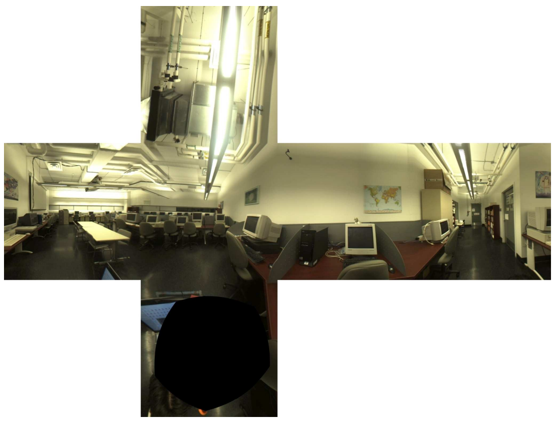

Figure 1. Example of a cubic panorama of the outdoor set laid out in a cross pattern.

3. Find more matches using SIFT with a relaxed threshold, and use the re-computed E to remove mismatches

4. Extract translation vector **t** (up to a scale) and rotation matrix **R** from SVD decomposition of the essential matrix **E**.

3 View Interpolation using Ray-tracing

Our goal is to generate photo-realistic novel views for virtual navigation. Given a set of pre-captured cubic panoramas, we are interested in generating arbitrary novel views so that we can achieve seamless visualization of environments from arbitrary viewing positions and orientations.

The approach used in this paper is straightforward and well-known in image synthesis: for every pixel of the target image, the optical ray coming from the optical center is traced in the 3D world. A search along this ray is therefore performed to locate the corresponding point in the reference images. This search is guided by a colour-consistency constraint. Once the correct 3D point is identified, the pixel colour in the target image is simply obtained through backward mapping.

Let us consider the example of the interpolation of one view from two reference cubic panoramas (see Figure 2). The two reference cubes are C_0 and C_1, and the world frame is attached to the frame of C_0. The Euclidean transformation between the world (cube C_0) and C_1 coordinate frames is specified by $\mathbf{R}_{01}$ and $\mathbf{T}_{01}$. We need to generate the novel view C_s, which has an Euclidean transformation $[\mathbf{R}_{0s}|\mathbf{T}_{0s}]$. For an image pixel $\mathbf{x}_s(i)$ of target cube C_s, we trace the optical ray $\mathbf{O}_s\mathbf{x}_s(i)$ in order to locate the 3D point $\mathbf{X}(i)$, that is the intersection of the ray with one object of the environment. If no occlusion is involved (cf. Section 4.3), $\mathbf{x}_0(i)$

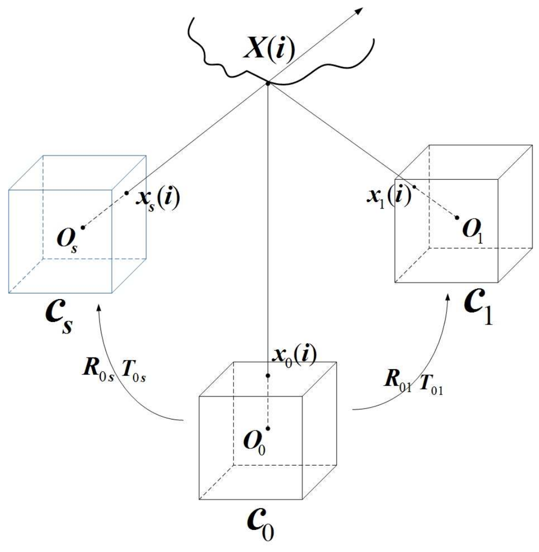

Figure 2. Novel view generation using ray-tracing.

and $\mathbf{x}_1(i)$, the projection of the 3D point $\mathbf{X}(i)$ onto cube C_0 and cube C_1 respectively, should have same colours. This colour consistency information is used to guide the search for the depth value of the 3D point $\mathbf{X}(i)$.

3.1 Colour consistency

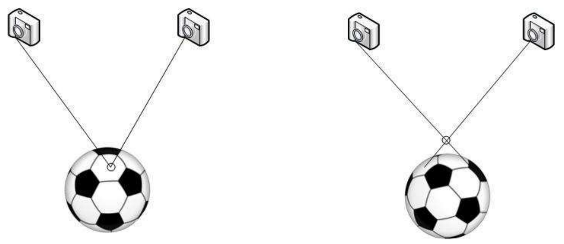

Figure 3. Colour consistency. On the left, the pixels from two images have the same colours at a point on a surface. On the right, the pixels from two images show inconsistent colours at points not on the same surface.

Colour consistency, introduced by Seitz et al. [14], is widely used in volumetric scene reconstruction [14, 1, 6, 2, 7]. As shown in Figure 3, it is used to differentiate surface points from others in a scene. It is assumed that the scene is completely composed of rigid Lambertian surfaces under constant illumination. If two pixels show inconsistent colours, they must be the projection of different scene points.

The colour consistency of a set of pixels can be defined as the maximum of absolute difference of colour channels between all pairs of pixels [2]. Let $\mathbf{X}$ be a 3D point, and $R_i(\mathbf{X})$, $G_i(\mathbf{X})$, $B_i(\mathbf{X})$ be the three colour channels of visual information at the projection of $\mathbf{X}$ on view *i*. Then we have

$$\begin{aligned} |\gamma_i R_i(\mathbf{X}) - \gamma_j R_j(\mathbf{X})| &\quad + \\ |\gamma_i G_i(\mathbf{X}) - \gamma_j G_j(\mathbf{X})| &\quad + \\ |\gamma_i B_i(\mathbf{X}) - \gamma_j B_j(\mathbf{X})| &\quad < \quad \Theta \end{aligned} \tag{1}$$

with

$$\gamma_i = 1/\left(R_i(\mathbf{X}) + G_i(\mathbf{X}) + B_i(\mathbf{X})\right). \tag{2}$$

The threshold Θ is applied to decide if the pixels are the projection of the same scene point $\mathbf{X}$. The normalization factor is introduced to reduce the effects of the illumination variations.

Another method to measure the colour consistency of a 3D point is by computing the standard deviation of its projected pixel colours [11, 7]. Let $\bar{R}(\mathbf{X})$, $\bar{G}(\mathbf{X})$, $\bar{B}(\mathbf{X})$ be the three colour channels of the visual information at $\mathbf{X}$ averaged over n views, we can compute the deviation of view i as:

$$\begin{aligned} d_i^2(\mathbf{X}) &= \left(R_i(\mathbf{X}) - \bar{R}(\mathbf{X})\right)^2 + \left(G_i(\mathbf{X}) - \bar{G}(\mathbf{X})\right)^2 \\ &+ \left(B_i(\mathbf{X}) - \bar{B}(\mathbf{X})\right)^2 \end{aligned} \tag{3}$$

This deviation will be low if all the cameras see the same surfaces point $\mathbf{X}$. Otherwise, cameras viewing different points of the scene will result in a large deviation.

4 Algorithm Implementations

4.1 Brute-force Depth Searching

A first direct approach consists in trying to determine the correct backward mapping through an exhaustive search. Given an image point of a novel view, we trace the optical ray $\mathbf{O}_s\mathbf{x}_s(i)$, expressed as

$$\mathbf{X}(i) = \lambda_s(i)\,\mathbf{x}_s(i) \tag{4}$$

where $\lambda_s(i)$ corresponds to the depth (up to scale) of 3D point $\mathbf{X}(i)$. This brute-force depth searching algorithm is given in Algorithm box 1. In our experiments, we set $\lambda_{min} = 0.0005$, $\lambda_{max} = 2$ (the depth is up to scale), $\epsilon = 0.5$ and `step` $= 0.0002$.

```
forall $x_s(i)$ do
    $d^{opt} = \infty$;
    for $\lambda_s(i) = \lambda_{min}$ to $\lambda_{max}$ do
        compute $\mathbf{X}(i) = \lambda_s(i)\,\mathbf{x}_s(i)$;
        project $\mathbf{X}(i)$ to cube $C_0$, $C_1$,...,$C_m$ ;
        compute $d_k(\mathbf{X}(i))$ using Equation (3);
        if $\sum d_k(\mathbf{X}(i)) < d^{opt}$ then
            $d^{opt} = \sum d_k(\mathbf{X}(i))$;
            $\lambda_s^{opt}(i) = \lambda_s(i)$;
            if $d^{opt} < \epsilon$ then break;
        end
        $\lambda_s(i) += step$;
    end
end
```

Algorithm 1: Brute-force Depth Searching Algorithm

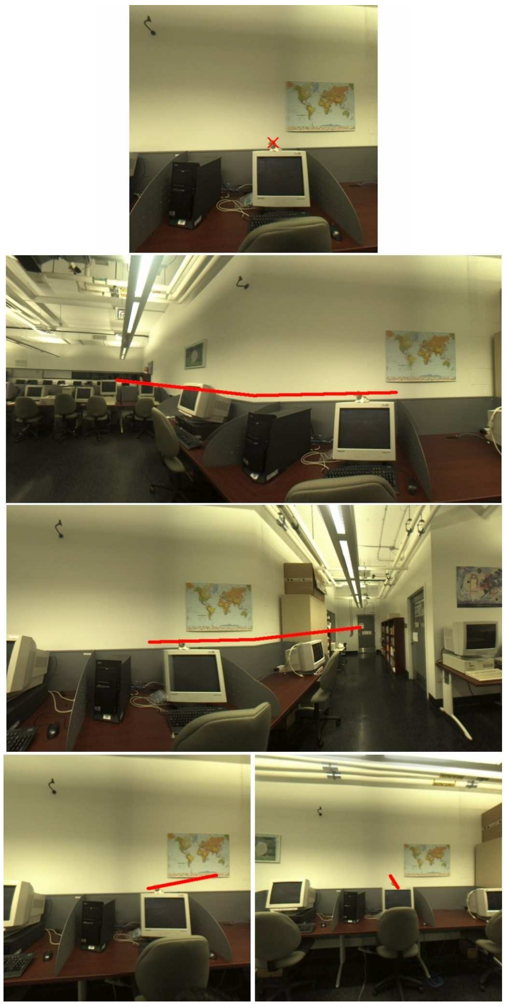

Figure 4. Brute-force depth searching. A point is chosen from "virtual" cube *s*. The corresponding search ranges are shown on the four reference cube faces of the set.

After the optimal depth $\lambda_s^{opt}(i)$ is found, we can back-project the point $\mathbf{X}(i) = \lambda_s^{opt}(i)\,\mathbf{x}_s(i)$ to all the input cubes in the set. This results in a set of projected image points, namely, $\mathbf{x}_0(i)$, $\mathbf{x}_1(i)$, ..., $\mathbf{x}_m(i)$ on reference cubes C_0, C_1, ..., C_m, respectively. The colour values of novel view pixel $\mathbf{x}_s(i)$ can then be interpolated from these input image points.

To illustrate the search range that is implied by this brute-force algorithm, we performed an experiment with five cubic panoramas. Four of them are used as references and the fifth one constitutes the ground truth for the panorama we will interpolate at its location (this fifth panorama is the one shown in Figure 1). We select one point on the right face of the target panorama as shown in Figure 4. The corresponding search ranges in the four reference panoramas are also shown in Figure 4 (we show here only the relevant cube faces). As it can be seen, these ranges can be quite large. This is especially true when an object is close to the camera, such as the searching ranges of cube *a* and cube *b* in Figure 4. The broader these search ranges are, the more "similar" colour pixels there can be in the reference cubes, and hence the higher the probability to find a wrong depth value. Of course, another fundemental problem with the brute-force strategy is its high computational cost.

4.2 Guided Depth Searching Through Sparse 3D Reconstruction

The view generation process can be improved by narrowing down the search range; this can be achieved by using the sparse matching results we obtained at the pose estimation step. Interestingly, it turns out that this modification improves both the speed of the view generation process and the quality of the interpolated images by reducing the chance of getting wrong depth values.

In this version of the algorithm, the search along the traced ray is guided by a sparse 3D reconstruction. Firstly, we find the pairwise sparse feature matches from the input cubes of the set. Then the corresponding 3D points are reconstructed. In order to get depth range information on the target view, we transfer the reconstructed 3D points from all input cubes onto the target cube frame.

The Euclidean transformation from the frame of cube C_i into that of cube C_j is denoted as $\mathbf{R}_{ji}$, $\mathbf{t}_{ji}$. The depth information of the j^{th} pixel $\mathbf{x}(j)$ of cube C_i is $\lambda_i(j)$. The 3D point corresponding to depth $\lambda_i(j)$ is $\mathbf{X}_i(j)$. Sparse reconstruction and point transfer then proceed as follows:

1. Use the method of Section 2 to find matches and compute $\mathbf{R}_{ij}$, $\mathbf{t}_{ij}$ for every two cubes of the set.

2. Given $\mathbf{x}_j(i) \longleftrightarrow \mathbf{x}_k(i)$, the i^{th} correspondence of cube C_j with the cube C_k, triangulate $\mathbf{X}_j(i)$, the 3D point (up to scale) in the frame of cube C_j.

3. Transfer all computed 3D points $\mathbf{X}_j(i)$ of cube C_j frame into cube C_0 frame (world frame) using:

$$\mathbf{X}_0(i) = \mathbf{R}_{0i}\mathbf{X}_j(i) + \mathbf{t}_{0i}. \quad (5)$$

4. Remove repeated 3D points from the set.

We thus obtained a large set of 3D points, designated $\tilde{\mathbf{X}}_0(k)$, that can then be projected on any novel viewpoint (a cube C_s). A set of image points $\tilde{\mathbf{x}}_s(k)$ and their depths $\tilde{\lambda}_s(k)$ in cube C_s is obtained. These depth values can be used to guide the searching during the ray-tracing procedure. The guided depth searching algorithm is given in Algorithm box 2.

```
project all $\tilde{\mathbf{X}}_0(k)$ onto $C_s$;
compute $\tilde{\mathbf{x}}_s(k)$ and $\tilde{\lambda}_s(k)$;
forall $\boldsymbol{x}_s(i)$ do
    $d^{opt} = \infty$;
    forall $\tilde{\lambda}_s(k)$ for which $\tilde{\mathbf{x}}_s(k) \in$ neighborhood of $\boldsymbol{x}_s(i)$ do
        compute $\mathbf{X}(i) = \tilde{\lambda}_s(k)\,\mathbf{x}_s(i)$;
        project $\mathbf{X}(i)$ to cube $C_0, C_1,...,C_m$ ;
        compute $d_k(\mathbf{X}(i))$ using Equation (3);
        if $\sum d_k(\boldsymbol{X}(i)) < d^{opt}$ then
            $d^{opt} = \sum d_k(\mathbf{X}(i))$;
            $\lambda_s^{opt}(i) = \tilde{\lambda}_s(k)$;
            if $d^{opt} < \epsilon$ then break;
        end
    end
end
```

Algorithm 2: Guided Depth Searching Algorithm

The experiment of the previous subsection is repeated for the guided search algorithm. The results are shown in Figure 5. Compared with Figure 4, it can be seen that the search ranges are much smaller than those of the *brute-force depth searching* which as it will be shown in the next section improve both the speed of the panorama generation and the quality of this interpolated panorama.

4.3 Dealing with Occlusions

All IBR systems must deal with the problem of occlusion. An occlusion occurs when a visible surface in some input images is hidden in another. To generate a novel pixel from a set of N reference cubes, ray-tracing is applied to find the depth for which the best colour consistency is obtained among the N reference panoramas. Then the reference cube with the pixel colour that differs the most from mean colour value is eliminated. The novel pixel is thus interpolated from $N - 1$ pixels.

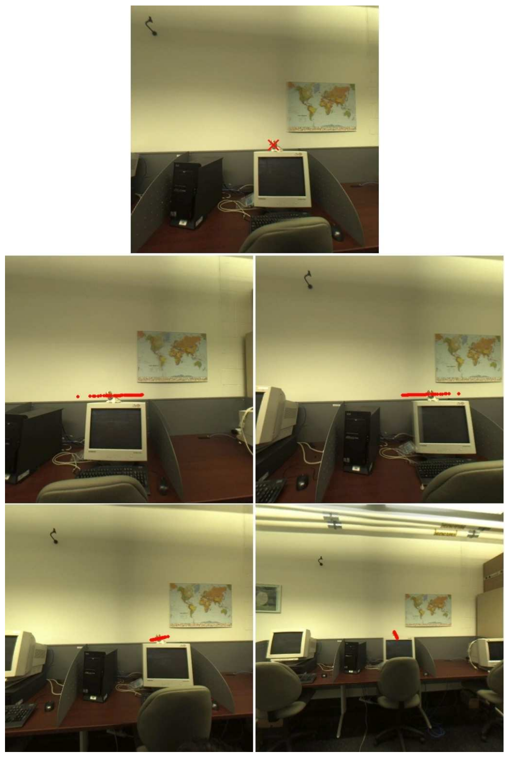

Figure 5. Guided depth searching. A point is chosen from "virtual" cube s. The corresponding search ranges are shown on the four reference cube faces of the set.

5 Experiments

An indoor and an outdoor experiments have been performed to test our panorama interpolation algorithm. We used 4 pre-captured cubic panoramas to generate a virtual cube plus one extra panorama that serves as *ground truth*. We therefore input the algorithm with the computed *rotation matrix* and *translation vector* of this extra cube. If our algorithm works well, the generated cube and the real cube should be identical.

For the first experiment, we used the four indoor cubes shown in Figure 6 as input views. The largest translation among these input cubes is about 1 meter. The world frame is attached with the frame of cube 1. We used the panorama shown in Figure 1 as our *ground truth*. The interpolated cubic panoramas generated from these four cubes using the two proposed implementations are shown in Figure 7. In the case of the guided depth searching algorithm (bottom of Figure 7), the results are good considering the complexity of the scene and the relatively large translation. However, reconstruction errors are visible, especially on the computer monitor on the right face. This monitor is very close to

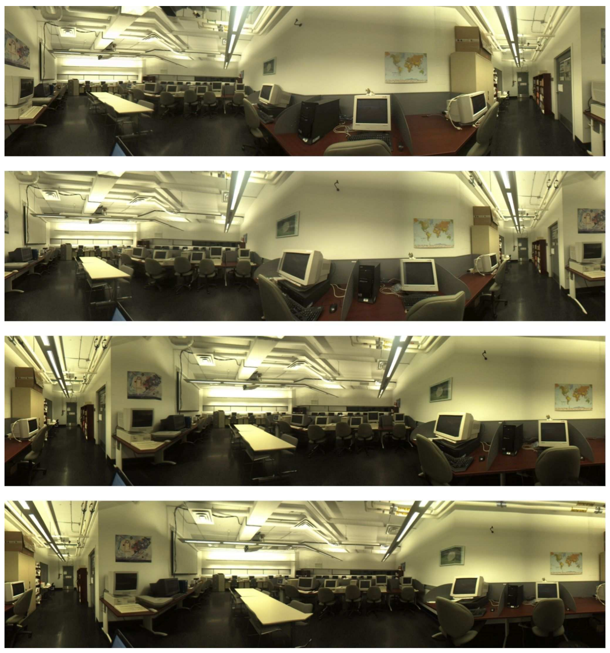
Figure 6. Indoor cube sequence (top and bottom faces not shown) used to generate virtual cubes.

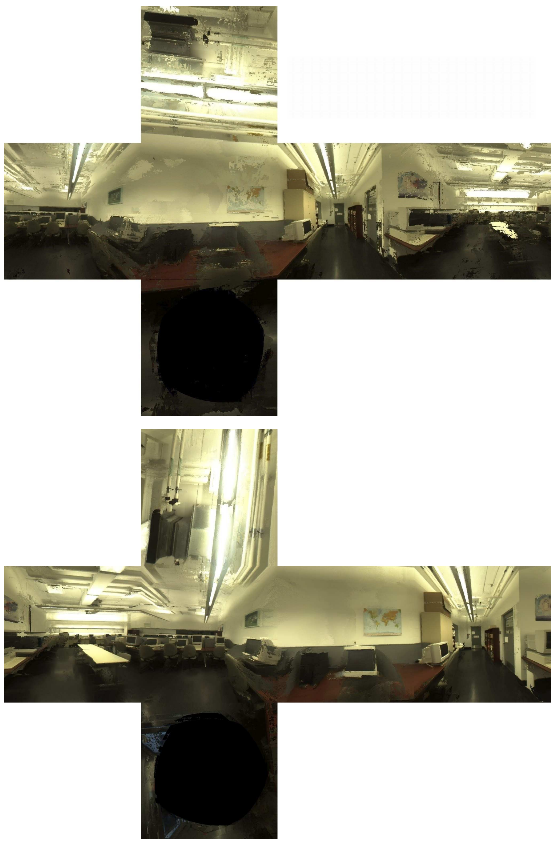
Figure 7. Virtual cube Vs. real cube for indoor cubes: the top cube is a virtual view generated from 4 cubes shown on Figure 6. This virtual cube is designated to produce the same view as the bottom real cube

camera, which results in a large searching range. As stated previously, the closer the objects are to camera, the broader the searching ranges, and the higher the probability is to reconstruct pixels with wrong colours.

The *guided depth searching* also improve the searching speed dramatically. To generate novel cube view from 4 input cubes, the computation time for a cube resolution of $512 \times 512 \times 6$ faces on an AMD 64x2 1.6GHz, 1024MB memory laptop computer took 26 hours and 33 minutes for the brute-force searching and 1 hour 23 minutes 38 seconds for the guided depth searching.

The next experiment shows the viewpoint interpolation results for a set of outdoor panoramas. In this case, the largest translation among the reference cubes is about 7 meters, which is a very severe condition for image interpolation. The four input outdoor cubes are shown on Figure 8, and the extra panorama used as *ground truth* is shown on top of Figure 9. The accuracy of the interpolation is not as good since the translation is very large. For example, there are very large reconstruction errors for the bikes in the scene. The main reason is that the large translations among input cubes result in very small resolution of bikes in *cube 6* and *cube 8* shown on Figure 8. Also, the homogeneous colours in the scene (the colours of the ground and the main building are almost same) put more challenges on our method.

6 Conclusion

This paper presented a comparison of two implementations of a ray-tracing algorithm for the interpolation of virtual viewpoints from a set of panoramic views. Our contribution was to show that using a very simple approach based on colour consistency leads to relatively good results. We showed also that by limiting the search space using sparse depth information improve both the speed and the accuracy of the interpolation. It should be emphasized that in our approach, we were concern only by the quality of the visual reconstruction; obtaining true depth information was not an objective here.

References

[1] W. B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. In *ICCV '99: Proceedings of the In-*

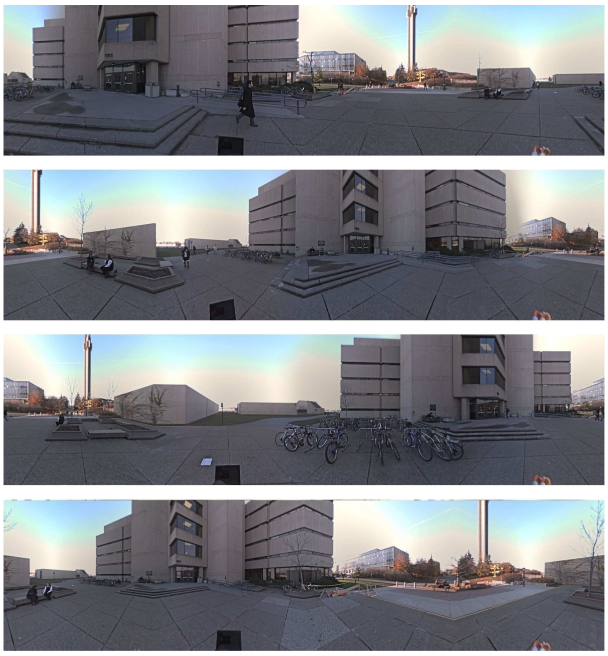

Figure 8. Outdoor cube sequence (top and bottom faces not shown) used to generate virtual cubes.

ternational Workshop on Vision Algorithms, pages 100–115, September 1999.

[2] P. Eisert, E. Steinbach, and B. Girod. Multi-hypothesis volumetric reconstruction of 3-D objects from multiple calibrated camera views. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP'99)*, pages 3509–3512, Phoenix, USA, 1999.

[3] M. Fiala. Immersive panoramic imagery. In *Canadian Conference on Computer and Robot Vision*, pages 386 – 391, 2005.

[4] N. Greene. Hierarchical polygon tiling with coverage masks. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 65–74, 1996.

[5] F. Kangni and R. Laganière. Epipolar geometry for the rectification of cubic panoramas. In *CRV '06: Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision*, page 70, 2006.

[6] K. N. Kutulakos and S. Seitz. What do N photographs tell us about 3D shape? In *Technical Report TR680, Computer Science Dept., U. Rochester*. January 1998.

[7] R. Laganière, H. Hajjdiab, and A. Mitiche. Visual reconstruction of ground plane obstacles in a sparse view robot environment. *Graphical Models*, 68(3):282–293, 2006.

[8] S. Laveau and O. Faugeras. 3-D scene representation as a collection of images and fundamental matrices. In *Proceedings of the 12th IAPR International Conference*, volume 1, pages 689–691, Oct. 1994.

[9] M. Lhuillier and L. Quan. Image interpolation by joint view triangulation. In *Proceedings of IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 139–145, Fort Collins, CO, USA, 1999.

[10] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Computer Graphics*, 29(Annual Conference Series):39–46, 1995.

[11] A. Prock and C. Dyer. Towards real-time voxel coloring. In *Proc. Image Understanding Workshop*, pages 315–321, 1998.

[12] P. Rademacher and G. Bishop. Multiple-center-of-projection images. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, volume 32, pages 199–206, 1998.

[13] S. Seitz and C. Dyer. View morphing. In *SIGGRAPH96*, pages 21–30, 1996.

[14] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. CVPR*, pages 1067– 1073, June 1997.

[15] J. Shade, S. Gortler, L. He, and R. Szeliski. Layered depth images. In *SIGGRAPH*, pages 231–242, July 1998.

[16] X. Sun and E. Dubois. View morphing and interpolation through triangulation. In *Proc. IS&T/SPIE Symposium on Electronic Imaging, Conf. on Image and Video Communications and Processing*, volume 5685, pages 513–521, San Jose, CA, Jan. 2005.

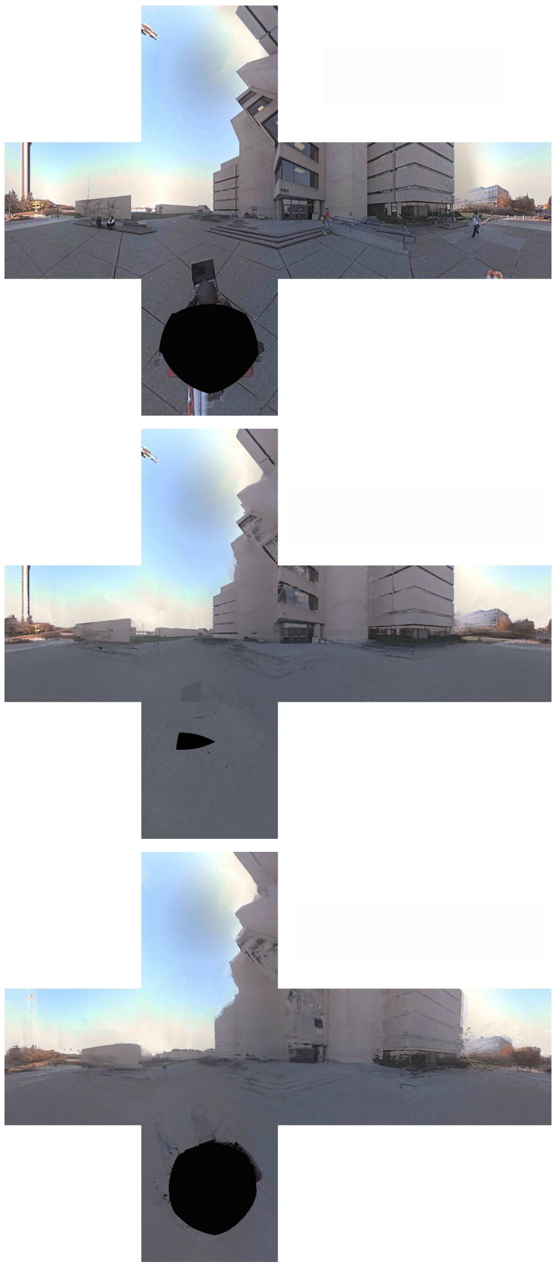

Figure 9. Outdoor panorama generation. Top: real captured panorama, center: panorama generated using guided depth searching, bottom: panorama generated using brute-force searching.