A Voxel-Based Approach for Virtual Objects Relighting

François Fouquet, Jean-Philippe Farrugia, Sylvain Brandel

Computer Graphics International 2011

A Voxel-Based Approach for Virtual Objects Relighting

François Fouquet · Jean-Philippe Farrugia · Sylvain Brandel

Abstract This paper presents a new way to insert virtual objects into real environments. It proposes an image based method to determine light interactions between the different parts of the scene. Unlike most previous works, our technique does not need any manual modeling: data are extracted from multiple 2.5D high dynamic range captures of the scene. These images are obtained via a calibrated time of flight camera coupled with an ordinary webcam. Visibility information and light transport are stored using a voxel grid. Using these data, we create an new image with a seamlessly inserted virtual object. We demonstrate the efficiency of our method by inserting a virtual object first in a virtual environment and next in a real environment.

Keywords Augmented Reality · Global Illumination · Relighting

1 Introduction

The purpose of this work is to automatically insert virtual objects into acquired views of a real scene (figure 1) and estimate light transport between real and virtual parts of the augmented scene, in accordance with following characteristics:

- we do not perform any offline computation and we do not need geometric modeling of the scene, in opposition with existing methods [3] [5] [6] [7];
- we manage short range illumination and occlusions between real and virtual parts, and automatically deal with long and near field illumination, in opposition with other methods [2] [9];

François Fouquet · Jean-Philippe Farrugia · Sylvain Brandel
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205, F-69622, France
Tel.: +33-4-26234448 Fax: +33-4-72431536
E-mail: francois.fouquet@liris.cnrs.fr

Fig. 1 An image of the real scene (left) is augmented by inserting a virtual object (right).

- no light probe nor hemispherical cameras are needed.

Our method is based on a custom device able to capture high dynamic range (HDR) depth and color images in near real time. We use data from this device to instantly construct an approximation of scene geometry by re-projecting pixels of the image plane into scene voxels. Local light transport is then modeled by tracing rays from high radiance voxels to the rest of the scene. When inserting a virtual object, one or more of these rays may bet cut, allowing us to compute shadows and virtual object illumination by differential rendering. We assume following:

- light sources have to be shown during acquisition process, and, for each point in space, the main part of incident light comes from a limited number of sources;
- only lambertian virtual objects and point sources are handled;
- inter reflections and color bleeding are not taken into account.

The second section will expose our algorithm while the third one will show some results, using virtual and real data. The last one will conclude with future works.

2 Image-based hybrid global illumination

We chose an intermediate approach between rendering and image-based techniques. The main idea is the same as in global illuminations acceleration methods: for a given point in space, the main part of incident light comes from a limited number of sources. Therefore, we may consider that this energy is a discrete sum of a small number of principal light streams, and neglect or globally evaluate the rest.

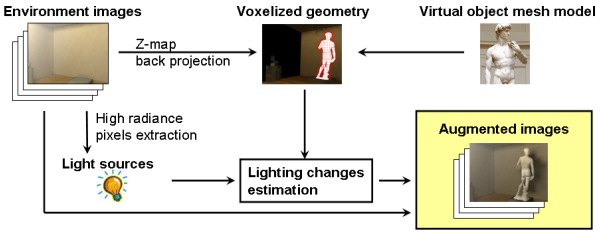

Fig. 2 Method overview.

Under this assumption, we do not need our environment representation to be precise enough to render a new view of the scene. It should only store enough geometry to compute occlusion and principal light exchanges between all parts of the scene. Hopefully, HDR images coming from calibrated cameras may be considered as a sampling of light exchanges in the scene. Gathering information from several depth and color images allows us to reconstruct geometry and lighting interactions between all parts of the scene. However, environment geometry is not known in advance. Our initial dataset is only composed of calibrated HDR images along with their corresponding depth map. Next sections describe our method, summarized on figure 2.

2.1 Geometry and light sampling

Input data are HDR with depth (HDR-Z) images. These images contain information of two types: coordinates of 3D geometry points in camera reference frame, with corresponding depth, and outgoing radiance associated to these points. Therefore we may consider that acquiring a big set of such images is very similar to sampling methods used for raytraced images. Assuming a dense enough sampling, we may transform and use these information to relight virtual objects with real environment.

We chose to store geometric information into a regular voxel grid, which is a simple structure easily up to date, each voxel is acceded only once. For each image, all HDR-Z pixels are backprojected into the voxelized space in scene reference frame. We use inverse of intrinsic and extrinsic parameters matrix to perform this backprojection. When several pixels reproject in the same voxel, a mean value of their radiance is stored.

At the end of this step, we obtain a set of voxel bouding the 2.5D surface observed on image. Repeating this operation on all input images gives us the voxelized surfaces of all the objects observed in the sampling set. Figure 3 left shows our rough voxel set, with a 64^3 resolution. Major part of the voxels contain data about the scene, so we do not need to deal with sparse matrix or compression schemes. The approximation in a voxel volume implies lack of precision and square shadows in rendered results, but satisfactory corrected with a 3D smoothing function.

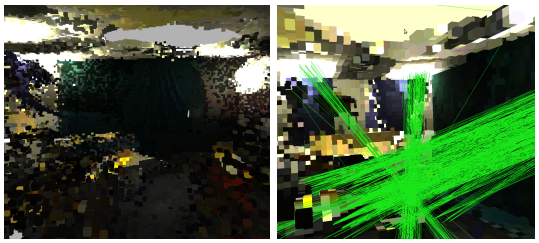

Fig. 3 Voxel set (left) with associated rays (right).

We made the assumption that a small number of light samples are enough for a plausible virtual object lighting. The first sampling step is the segmentation of the HDR input image. We use a fixed threshold and separate pixels into two groups: pixels having a high radiance (above the threshold) and pixels having low radiance. Notice that we use the same threshold for all images of the set in order to have coherent classification for the whole scene. Then, pixels are processed according to their radiance level:

- For each *high radiance pixel*, we create a ray in voxel grid space. This ray starts on the visible 3D point obtained by backprojecting the pixel. It points towards the camera and stores the pixel radiance as intensity value. All observed rays are finally gathered in a ray list attached to the scene (figure 3).
- Radiances of *low radiance pixels* are averaged on all images. The computed value is an estimation of ambient energy that is not considered in any ray.

Rays are only used for shading and shadowing virtual objects, so there is no other ray for global illumination.

2.2 Virtual object insertion

We first insert the virtual geometry in the voxel space. Then we can identify regions where lighting conditions are modified. For each ray, we trace it into the voxel space using a voxel traversal method [1]. If the ray intersects the virtual geometry, the first virtual voxel is stored and lit using the ray intensity. We also store the first environment voxel intersecting the ray, as the energy of the intersected ray do not reach it anymore. After this step, the structure is ready for computing relighted images of the scene including virtual object.

2.3 Rendering augmented images

Once the voxel grid is updated, we compute two images. A primal image containing only virtual objects is rendered. Vertex colors are set and shaded using lighting information of their corresponding voxels and normals. A second image of light difference coefficients is computed by backprojecting voxel coefficients on image plane. It is then multiplied with the input image to have an image of environment influenced by the virtual object. Both images are then combined according to their depth map to form an augmented image containing virtual elements and shadowed environment.

3 Applications

For testing purposes, a purely virtual scene was first used. This scene represents a museum room and was sampled the exact same way it is described in the previous sections. The virtual object is then inserted in this scene using only the HDR and depth captures made on the virtual scene. Comparisons are made between the result and the entirely ray-traced scene on figure 4. Despite some surface artifacts and shading differences between our renderer and the reference ray tracing engine, one may see that shadows and shading are similar.

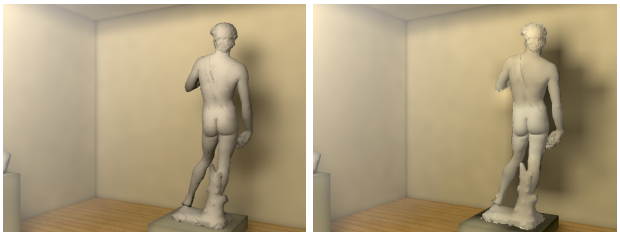

Fig. 4 Insertion of a David statue in a museum virtual scene. Left: ray-traced reference picture. Right: our method.

A second application illustrates the whole process, with the acquisition of a real environment and the insertion of the same virtual David statue in this environment. The test scene is a closed classroom in which we arranged a free space to allow camera movement. We placed three light sources in this room: one of them is directly visible by the camera while the two others are not, generating only indirect lighting. While these lights are placed by hand, we do not track them and our method does not use their position. The scene is divided into two parts. In the first one, a few real objects are present: a teapot, a chair and a small table. The camera may move freely in this part. The second part is uncontrolled and consists of desks, computers, screens, and various furnitures (Figure 5).

Captured data is composed of HDR images and corresponding depth images. The Z channel is obtained using a Mesa time of flight camera (TOF) SR4000. The information coded on this channel is the distance of pixel object to the camera position. Since the TOF camera does not provide HDR color image, we coupled it with a Logitech Pro 9000 webcam to record low dynamic range (LDR) color image (see figure 6). The low resolution of the webcam is enough because SR4000 provides a lower resolution. As described in [4], we use the shutter time variation ability of this device to produce three LDR image with different exposures, then we combine these images to obtain an HDR image. Both cameras are stabilized with a tripod. To achieve localization, the tripod wears a fiducial pattern which allows a third camera, located on the ceiling, to track its position. A second fiducial pattern is laying on the ground and represents our reference set. We use Open CV Calibration procedures to obtain extrinsic and intrinsic parameters for all the cameras in the scene.

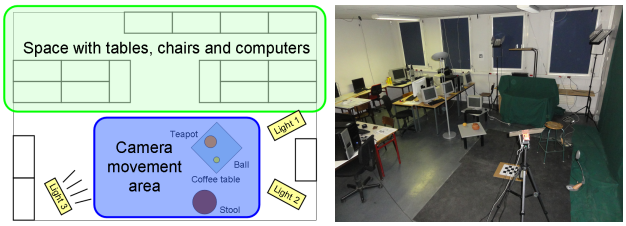

Fig. 5 Plan and global view of our test scene.

Fig. 6 Our hardware setting.

After this acquisition, the captured HDR and Z image have different sizes and points of view. To compute the HDR-Z image, we project the HDR pixel on the right Z pixel in TOF camera image. To facilitate this reprojection, both cameras are rigidly fixed together and registered with the same chessboard fiducial pattern. Using calibration information, our program computes the transform matrix used for this projection. To ensure that all Z pixel will have a corresponding color, the webcam field of view is set greater than the SR4000 one and cameras are adjusted in a way that the TOF camera's viewing frustum is included in the webcam one. The HDR image size is also bigger than Z image size in order to avoid multiple projection of the same color on different Z pixels. The final image has the same point of view and size (176×144) than the input Z image.

170 view are captured, from various point of views, with a preference for views around light sources. Images

are in EXR format, with 7 channels: R, G, B, X, Y, Z, confidence. The XYZ encodes the position of the pixel in the scene reference frame, it is computed from the z data captured by the device. When all the captures are taken into account and reprojected, a total of 120 476 rays are traced in a $64 \times 64 \times 64$ voxel grid.

To perform these tests, we used a 2.83 GHz Core 2 quad desktop PC equipped with a nVidia Geforce 9600 GT graphics processor. Acquiring an image is around 300ms. Constructing / updating octree for real and virtual data is 40ms. Tracing and updating rays is 220ms for all rays. Cleaning up noisy voxels is 20ms. Rendering final image is 120ms. Of course these computing times depend on the size of the voxel grid and consequently on the number of traced rays.

With this dataset and the previously described method, we may produce a model of light transport through the scene and use it to insert a David statue in our classroom. The depth maps are median filtered to eliminate pixels with a low confidence score. Result from various point of views are shown on figure 1 and figure 7. Of course it is possible to navigate anywhere in the scene and show all the faces of the statue. The statue's shadow is correctly projected on the floor, with a blur shadow coherent with indirect illumination. Moreover, the statue is in a correct spatial position, occluding farther real objects. The blurriness of the pictures may be explained by the low resolution of the output.

Fig. 7 Results of our method: original HDR image (left) and augmented image (right).

4 Conclusion and future work

In this paper, we demonstrated a fully automatic method to insert a virtual object into a real scene with correct lighting. A specific device has been assembled and used to capture environment data with minimal calibration. The whole process is computed at an interactive rate and does not require manual geometric modeling.

A lot of improvements are yet to do with this method. First, the HDR-Z device could be made of consumer hardware if we replace the MESA SR-4000 camera with the significantly cheaper Kinect Xbox 360 camera from Microsoft, which basically have the same functionalities. We may also simplify it by only using webcam with real time depth acquisition techniques [4] [8]. As said in previous section, if the HDR-Z device were faster, an interactive reconstruction process may be possible.

The method may also extend quite easily to dynamic environments by allowing temporal changes of the radiance stored in the voxels. It may handle non diffuse and animated objects by taking into account non-constant reflectance factors. Of course, this implies a more complex rendering algorithm like ray-traced shading. In this case, we may think about other ways to model light transport. Importance sampling point-like sources from an emissive map may be a good start.

Acknowledgements This work was supported by the French National Research Agency, ref. ANR-07-MDCO-001.

References

1. Amanatides, J., Woo, A.: A fast voxel traversal algorithm for ray tracing. In: In Eurographics 87, pp. 3–10 (1987)
2. Cossairt, O., Nayar, S., Ramamoorthi, R.: Light field transfer: global illumination between real and synthetic objects. ACM Transactions on Graphics **27**(3), 51–57 (2008)
3. Debevec, P.: Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. Computer Graphics **32**(Annual Conference Series), 189–198 (1998)
4. Fouquet, F., Farrugia, J.P., Michoud, B., Brandel, S.: Fast Environment Extraction for Lighting and Occlusion of Virtual Objects in Real Scenes. In: IEEE International Workshop on Multimedia Signal Processing (2010)
5. Fournier, A., Gunawan, A.S., Romanzin, C.: Common illumination between real and computer generated scenes. In: Proceedings of Graphics Interface '93, pp. 254–262. Canadian Information Processing Society, Toronto, Ontario, Canada (1993)
6. Gibson, S., Cook, J., Howard, T., Hubbold, R.: Rapid shadow generation in real-world lighting environments. In: EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering, pp. 219–229. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2003)
7. Sato, I., Hayashida, M., Kai, F., Sato, Y., , Ikeuchi, K.: Fast image synthesis of virtual objects in a real scene with natural shadings. Systems and Computers in Japan **36**(14), 102–111 (2005)
8. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vision **47**(1-3), 7–42 (2002)
9. Unger, J., Gustavson, S., Larsson, P., Ynnerman, A.: Free form incident light fields. In: EGSR'08: Proceedings of the 19th Eurographics symposium on Rendering. Eurographics Association (2008)