Occlusion Handling and Image-based Lighting using Sliced Images in 3D Photo Collections

Frank Nagl* Konrad Kölzer† Paul Grimm‡

Fulda University of Applied Sciences, Germany

Figure 1: (a) 3D Photo Collection with embedded object, (b) Close-up of unlit object with wrong occlusion. (c) Photo divided into 3D segments (Sliced Image). (d) Convincing augmentation with our occlusion handling and image-based lighting features.

ABSTRACT

This paper presents novel methods to handle potential occlusion problems and to render realistically the lighting of embedded virtual 3D objects in photo-based 3D worlds. These 3D worlds are called 3D Photo Collections. The first focus of this work handles potential occlusion problems between image parts of photos and the embedded virtual objects by dividing photos into 3D segments, which we call *slices*. The segmentation of the photos is done by a gradient-based contour tracing algorithm, which divides a photo into several contours. These contours are used in the 3D world as slices. Each slice will be moved to a depth position estimated by the provided spatial interdependency between photos of the 3D Photo Collection. This results in several slices at different depth positions per photo. We call the divided photo *SlicedImage*. The second focus of this paper is concentrated on the realistic rendering of the illumination of embedded virtual objects by extracting the environmental lighting of the real scene from the generated Sliced Images.

For fully automated generating of image-based 3D worlds, 3D Photo Collections (generated by software like Microsoft Photosynth or Google Street View) are used. These 3D Photo Collections provide spatial interdependency between photos, which will be used as input data for our algorithms to compute the Sliced Images.

Keywords: 3D Photo Collection, Occlusion, Sliced Image, Depth Map, Image-based Lighting, Augmented Reality

Index Terms: H.5.1 [Information Systems]: Multimedia Information Systems—Artificial, augmented, and virtual realities

1 INTRODUCTION

Augmented Reality can be used in applications of urban planning and interior design to combine virtual products and real environments for a convincing impression. A well-established way of Augmented Reality is the augmentation of photos. In contrast to the usage of a single photo, an image-based 3D world can be built by shaping the real environment and matching a set of photos into it. This assumes the scene geometry measures and good experience with Photoshop or 3D modeling tools as well as a high effort to build an image-based Augmented Reality world. Another way is the usage of a structure-from-motion software for unordered image collections like Photosynth [20] or Google Street View [15]. A Structure-from-motion tool processes a set of unordered images and automatically provides intrinsic and extrinsic camera parameters as well as a sparse 3D model of the real scene as a key point cloud. The outcome of this is an image-based 3D world, which we call 3D Photo Collection. Information about the scene geometry or good experiences with special 3D modeling tools are not necessary. Also, arbitrary cameras are usable instead of a specialized hardware (e.g. see-through displays or tracking devices).

Owing to these advantages, our work uses 3D Photo Collections as base AR environment. In these 3D Photo Collections virtual 3D objects will be embedded. In this connection the visual appearance of virtual portions needs to be very authentic for the user to ensure a convincing AR impression. This requires to deal with a number of typical open issues in AR environments like

- occlusion problems of virtual parts by real parts, and
- illumination of virtual parts consistent with real parts.

This paper is focused on handling potential occlusion problems between image parts of the photos and the embedded virtual objects by dividing photos into 3D segments, which we call slices. Each slice will be moved to a depth position estimated by the provided spatial registration between photos. This results in several slices at different depth positions per photo. We call the divided photo Sliced Image. The second focus of this paper is concentrated on the realistic rendering of the illumination of embedded virtual objects by extracting the environmental lighting of the real scene from the generated Sliced Images. For lighting purposes, the images of the 3D Photo Collection should be shot with varying light exposure.

This paper is organized into 5 chapters. The next chapter is engaged with related works to this paper. Chapter 3 describes the concept of occlusion handling and image-based lighting of virtual products in 3D Photo Collections. After that, chapter 4 discusses the implementation as well as our results. The last chapter summarizes the content of this paper and gives some information about potential future work topics.

*e-mail: frank.nagl@hs-fulda.de
†e-mail: konrad.koelzer@hs-fulda.de
‡e-mail: paul.grimm@hs-fulda.de

The 21st International Conference on Artificial Reality and Telexistence
November 28-30, 2011, Osaka, Japan
ISSN: 1345-1278

2 RELATED WORK

This chapter is separated into the three issues

- 3D Photo Collections,
- occlusion handling, and
- image-based lighting.

Generating **3D Photo Collections** from unordered image sets is automatically performed by structure-from-motion algorithm tools like Bundler[29] or Photosynth [20]. Bundler first calculates image features for each photo using the SIFT algorithm[19] and then performs a pairwise image matching. After that, the intrinsic and extrinsic camera parameters are reconstructed for each image by using a modified version of the Sparse Bundle Adjustment package of Lourakis and Argyros[18]. The output of Bundler also contains sparse scene geometry as a set of 3D key points. Additionally, for each key point Bundler determines a color and the set of cameras that use this point as image feature. Based on Bundler, our IP3D framework was presented in [22]. This framework provides a generic architecture for fast and robust development of applications using 3D Photo Collections to build authentic augmented 3D worlds. Also in this work, the IP3D framework is used as the underlining 3D Photo Collection viewer.

In [14] it is shown how the spatial data structures of 3D Photo Collections will be used to realize a ambient view interpolation of foreground objects shown in several photos [14].

Several approaches for **handling occlusion** issues in the context of Augmented Reality with 3D Photo Collections exist. The patch-based multi view stereo algorithm of [12] and the 3D reconstruction algorithm of [30] show how 3D Photo Collections can be used to reconstruct 3D geometry from objects, which are depicted on many photos. These 3D objects handle possible occlusions of other virtual objects in an augmented 3D Photo Collection. The requirements for these works are a large quantity of photos, the foreground objects have to be emphasized considerably from the background in the photos and the algorithms do not accept arbitrary photo scenes. In conclusion, these approaches are too restrictive for our proposed ideas.

A depth map can be used to extract foreground and background objects in photos. Further work dealing with estimating depth maps from photos is discussed in [3]. A foreground object in single photos is extracted by estimating a depth map. Therefore, a special lens aperture with an integrated RGB filter is used to encode three different grayscale images into one three-channel photo. Another work [31] estimates the depth positions of foreground objects in a video stream by using a stereo view algorithm. Two exact calibrated cameras are needed for the stereo effect. Both approaches do not work with arbitrary photos and require special hardware. As a consequence, they cannot be used in our approach. In our poster [23], we presented a first approach for generating Sliced Images by using 3D Photo Collections. Every 3D key point is projected into the photo and the distance between camera position and key point is used for the color value of the pixel. Every 2D projected key point is used as cell nucleus for a Voronoi Diagram [1]. A depth map will be generated by applying the Voronoi diagram. In contrast to this first approach, our new approach in this paper provides an improved version of generating the depth map for building the slices. Instead a Voronoi diagram, a contour tracing algorithm divides the photos into segments. 3D key points which belong to a segment are used for estimation the depth of this segment (see section 3.1).

The next approaches deal with the issue **image-based lighting**. Fournier et. al. [11] presented differential rendering to embed virtual objects with correct shadows into a real scene. This is done by performing an approximate manual reconstruction of geometry, camera parameters and lighting conditions of the real scene.

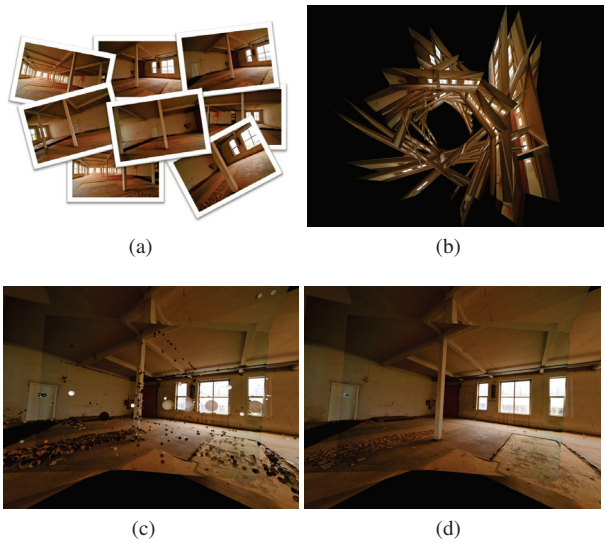

Figure 2: (a) Unsorted photos. (b) Matching photos together to a 3D Photo Collection, seen from top view. (c) 3D Photo Collection with key points. (d) Generated 3D Photo Collection, seen from front view.

Then the scene is rendered once without and once with virtual objects. The difference is then added to the photo to produce realistic shadows. Another approach was presented by Debevec in [9]. Debevec divides the real environment into distant and local scene. Light sources and geometry of the local scene are manually approximated. The lighting of the distant scene is described as a high dynamic range 'omni-directional Radiance-Map', which is provided by a measured light probe. The usage of light probes was later adapted to real-time rendering, by extracting few directional light sources from the environment map [10, 8].

A novel approach of Gibson [13] also uses light probes, which are projected on the coarsely reconstructed geometry of a real scene. The geometry is subdivided into patches and Inverse Radiosity is used to calculate the diffuse factors for each patch. Additionally, an Irradiance-Volume is precalculated that stores the environmental lighting for each point on a 3D grid as spherical harmonics coefficients and is used to perform real-time capable lighting. This was later extended by Grosch in [16] by generating shadows from direct illumination. Both approaches give good results with realistic shadows, however they also share the restriction to panorama scenes: The light of the real scene is measured by a light probe, which only specifies lighting information for scene parts that are not occluded from the viewing position of the light probe.

In [17] and [23], we presented another approach for image-based lighting by using 3D Photo Collections. Each pixel of an image is treated as a ray of light that is either emitted or reflected from a point on a surface of the real scene. In contrast to this paper a much simpler depth reconstruction of the image pixels is used: For each photo a single depth value is estimated and used for all pixels which is a extremely rough reconstruction.

3 APPROACH

The main goal of our work is to increase the visual quality by addressing occlusion problems and image-based illumination in augmented 3D Photo Collections for convincing urban planning and interior design. A 3D Photo Collections viewer is used to visualize the reconstructed real environment (see fig. 2) and 3D geometries are given for embedding virtual objects.

To address the problems of our main goal, the viewer has to provide features for occlusion handling and for image-based lighting.

These features will be described in the following subsections.

3.1 Occlusion Handling

Using augmented photo collections some image parts of photos may occlude parts of the embedded virtual objects and vice versa. Since occlusions are essential for human visual perception [4][2] and a faulty visualization or an incorrect occlusion handling destroys the Augmented Reality experience [25], occlusion handling in AR applications is a fundamental aspect. Therefore, a photo has to be divided in separated regions. Next, the correct rendering order of foreground image parts, virtual objects and background image parts has to be considered. Thus, three issues are important:

- Separation of foreground and background image parts,
- estimation of 3D depth position of separated image parts, and
- occlusion of embedded virtual objects by image parts.

Separation of foreground and background image parts The photos have to be divided and segmented. For this purpose we use a novel contour tracing mechanism. In contrast to other contour tracing and edge detection algorithms our contour tracer results in closed contours with the knowledge which pixel belongs to which contour. We presented in [24] a previous version of this contour detector. For better clarity of the overall process, we describe the contour detector in detail.

By applying an edge detection to the image (converted in gray scale), the gradient and consequentially the orientation are known for each pixel [7]. Based on this knowledge a three-stage tracing algorithm is applied. For each pixel its neighbor pixels will be analyzed in a specified radius, if they are

1. edge pixels, when this returns no results then
2. own contour pixels, when this returns no results then
3. other contour pixels.

The analysis algorithm for all three steps searches for the neighbor with the lowermost match value:

```
if (Neighbor has same orientation)
  match = Max(distanceX, distanceY)
else
  match = Max(distanceX, distanceY) + weight
```

The lower the *match* value is the higher the significance of the neighbor for the contour is. The parameters *distanceX* and *distanceY* represent the distances between the pixel and its neighbor on the x- and y-axis in pixel coordinates. For calculating *match* the bigger value of the distances in x- and y-direction is used instead the euclidean distance. This is done to fill the match value matrix with integers and these computations are more efficient with the same result. Parameter *weight* is a specified constant value to privilege neighbors with same orientation. The higher this value is set the higher is privilege of neighbors with same orientation. This value is not set automatically, because different requirements at the content of photos are controllable by setting this parameter. For example a photos of buildings with many squared objects should get a high *weight* value in contrast to a photo with rolling contours. So the user has an influence on the algorithm by controlling this parameter manually.

Following example illustrates the analysis results for a red line in an image. The pixel p belongs to a diagonal red line and is the current pixel to analyze. In this example the *weight* constant is set to 5 to privilege very strong neighbor pixels with same orientation.

The resulting match value matrix shows that the direct neighbor top right has the lowermost match value ($match = 1$) and hence the

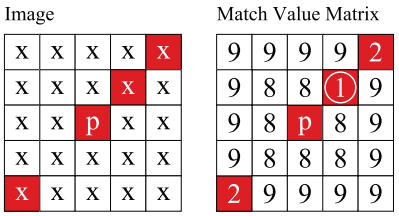

Figure 3: Illustration of calculated match values: Initial image with a diagonal red line and the corresponding match value matrix.

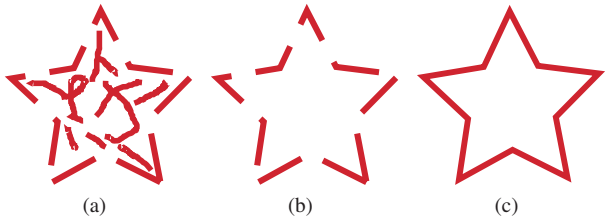

Figure 4: Getting a closed contour: (a) All pixels of one contour after analysis algorithm. (b) Detected borderline of the contour. (c) All borderline pixels connected by a polyline to a closed contour.

biggest significance. Consequential this neighbor becomes part of the contour of pixel p and the analysis starts from the neighbor's position again. This recursive process is done until all edge pixels belong to a contour.

The next step removes all contour pixels which do not belong to the contour's borderline (see fig. 4(b)). This is done by detecting the first and the last contour pixel per line as well as per column of the image. Finally, all detected pixels will be connected and drawn as a polyline [6] (see fig. 4(c)). This results in closed contours with the knowledge which pixel belongs to which contour. Fig. 4 shows the several steps to get a closed contour of a star-shaped object.

Estimation of 3D depth position of separated image parts After contour tracing, photos divided into image parts (contour segments) are given. The next step estimates the correct 3D depth position for every segment. The 3D depth position of a segment is defined by the distance between the segment and the 3D camera position of the segmented photo. The resulting values will be stored and visualized in a depth map.

Our approach for estimating the depth map uses all 3D key points which belong to a segment and calculates their *distances* to the corresponding 3D camera position. For evaluating, which 3D key points belong to a 2D segment, all 3D key points have to be projected into the 2D photos. Fig. 5 visualizes this projection and shows their *distances* to the 3D camera position.

With the knowledge of 2D image coordinates of the projected key points combined with the knowledge of each key point image coordinate belongs to which contour segment, the depth values of a segment can be estimated. Starting from each key point image coordinate a 2D flood-fill process is started until the fill area touches the segment's borderline or another fill area of a further key point. Thus, every key point produces a flood-fill area inside a contour.

Fig. 6 shows a draft with all key points related to a segment for the distance calculation.

Occlusion of embedded virtual objects by image parts The depth position of every image part (contour segment) is known. To answer the question, how image parts occlude the virtual objects, the image parts have to be moved to their calculated depth. This results in many slices per image, which we call **Sliced Image**. Fig. 7 shows the construction of a Sliced Image.

After considering occlusions, the next section deals with the image-based lighting of virtual objects in 3D Photo Collections.

3.2 Image-based Lighting

The generated slices of each image are a very sparse representation of the real scene's geometry. In this section we discuss how they are used to estimate the lighting conditions of the real scene.

Looking at the given data, image pixels can be identified as the smallest piece of available information. As shown in fig. 7, each image pixel provides an RGB color value, an incident direction (derived from its known camera view frustum) and a roughly estimated depth value. The challenge is to combine all this information from all image pixels to a meaningful representation of light conditions of the real scene.

Collect lighting information From now on collecting the environmental lighting of the scene shall be simplified to collect lighting that is received by a point of interest in the scene. For small sized virtual objects, the environmental lighting can be collected only for a single point (usually the object center). For larger objects, light can be collected from multiple points around the virtual objects. The latter case gives better results, but the extracted lighting has to be interpolated between points for final rendering.

The color value of an image pixel represents a certain amount of radiance that was reflected by a surface of the real scene onto the CCD chip of a digital camera. With the known direction and the estimated depth of a pixel, this point on the surface can be located roughly.

The scene surfaces are considered as diffuse (which is true for most surfaces in indoor scenes), so each pixel of an image does not only provide information for its camera position, but also for other positions in the scene, as long as it is not occluded by other surfaces of the scene.

From the above observations a colored ray can be constructed for each located surface point that is visible for the point of interest. Each ray gets the RGB color of its associated image pixel. The set of colored rays represent all available environmental lighting information for the point of interest.

Recovering relative radiance The RGB color of each ray cannot be used for lighting, because it is subject to clamping errors – high radiance is clipped to 100% white – and image noise – low radiance results in a more or less visible grain depending on the quality and the settings of the digital camera. Also the RGB intensities are not linear to the original radiance that was received by the camera lens due to camera specific non-linear transformations.

However, the color information can be linearized and extended to a high dynamic range relative radiance value, because the same point of a surface is usually shown on multiple images, which are

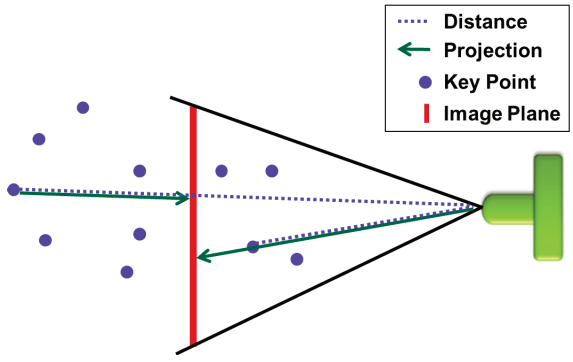

Figure 5: Projection of 3D key points to 2D image coordinates with their distances to the corresponding 3D camera position as pixel values.

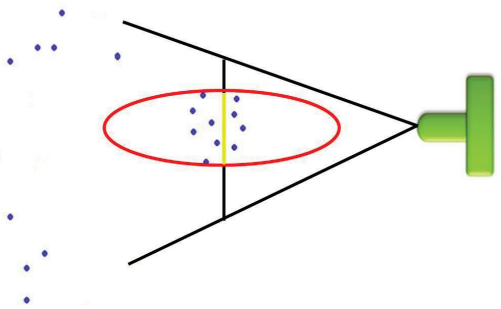

Figure 6: Red encircled 3D key points belong to the yellow 2D segment.

shot with varying light exposure. In other words the ray set contains many rays with identical direction, but varying RGB color.

The linearization is done by estimating the camera response curve using Robertson's algorithm [26]. Extending multiple rays to a ray with a relative radiance value can be calculated by adapting Robertson's HDR image estimate from image pixels to colored rays.

It is calculated as the weighted average of each ray r that shares the same direction. Each ray has the color value y_r the exposure time t_i originating from its associated image. The weighting function w returns the significance of color value and the camera response curve I linearizes the color value:

$$r_p = \frac{\sum_r w_{y_r} t_i I_{y_r}}{\sum_r w_{y_r} t_i^2} \quad (1)$$

Using this equation, a new set of rays is calculated that represents the known incident relative radiance for the point of interest.

4 REALIZATION AND RESULTS

To implement a 3D Photo Collection viewer, which provides mechanisms for occlusion handling and image-based lighting of virtual objects several toolkits are used. The structure-from-motion software Bundler [27] is used to extract the photos viewpoint (intrinsic and extrinsic camera parameters) of all images in the photo collection. Furthermore, a point cloud with all 3D key points is extracted, which is used to calculate the distance from camera to the image plane. The IP3D framework [22] renders the photos and key points in correct 3D orientation and position. For processing images the SBIP framework [21] is used for GPU-based performance improvements.

The solutions of our concept for a convincing augmentation are implemented in our 3D Photo Collection viewer. Fig. 8 shows

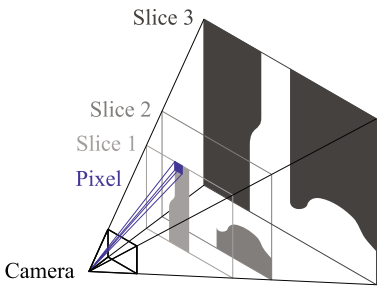

Figure 7: Pixel of a slice with known color, depth and incident direction.

(a)

(b)

Figure 8: A 3D Photo Collection with an embedded virtual chair: (a) Default augmentation without increasing the visual quality. (b) Augmentation with our occlusion handling and image-based lighting features.

the comparison of a 3D Photo Collection with an embedded virtual chair by standard augmentation as well as augmentation with our features. Fig. 9 shows the same scene with the virtual chair from a closer range to emphasize our results. The following subsections describe the implementation features and discuss the results.

4.1 Occlusion Handling Feature

The occlusion feature realizes a correct occlusion handling for every camera view (see comparison in fig. 9). For separating foreground and background image parts, a novel contour tracing algorithm is implemented, as described in section 3.1. The canny edge detector [5] is used for edge detection. The resulting contour segments are stored with all necessary information like pixel coordinates of their borderlines. Fig. 11 shows the result of our contour tracer.

The next step calculates the depth map (see fig. 12). The distances between the 3D key points related to segments and the 3D camera position are stored as depth values for the segments. For determining the relation of key points to segments, all 3D key points are projected on the 2D photos. For each camera the focal length (f), the rotation matrix (R) and the translation vector (t) is given from the IP3D framework. The following equations projects a 3D

(a)

(b)

Figure 9: Picture detail of fig. 8 with the virtual chair from a closer range: Comparison again (a) without and (b) with our occlusion handling and image-based lighting features.

key Point (X) into 2D pixel coordinate (p'):

$$P = R \cdot X + t \tag{2}$$

$$p = \frac{-P}{P.z} \tag{3}$$

$$p' = f \cdot p \tag{4}$$

(2) is to convert from world to camera coordinates, (3) represents the perspective division and (4) is the conversion to pixel coordinates [28]. In the current implementation a reviewing process for potential consolidation of segments by several conditions is not realized yet.

Finally, for occluding parts of the embedded virtual objects by image parts of photos and so handling the correct occlusion, the

Figure 10: Sliced Image occludes partial a virtual chair by a pillar in a 3D Photo Collection (seen from top view). Every slice is drawn with a white border to emphasize the multiple image planes.

(a) (b)

Figure 11: Our contour tracer: (a) Original image. (b) Contour image with colored segments to emphasize the result.

Sliced Image will be generated and rendered. This process is realized by using shader programs to ensure a real-time capable rendering of the scene. After contour tracing and generating the corresponding depth map of an image, a collection of contour segments and the correct depth position of every segment is given. For rendering the Sliced Image, a new image (slice) for every segment is generated. In this slice only the corresponding segment is opaque, all other image parts are transparent. Fig. 10 shows a Sliced Image and a partial occluded virtual chair in a 3D Photo Collection.

The next subsection describes the implemented image-based lighting using the Sliced Images and discusses the results.

4.2 Image-Based Lighting Feature

The image-based lighting feature applies a realistic illumination on embedded virtual object (see comparison in fig. 8 and 9). As well as the rendering of Sliced Images, the collection of environmental lighting is implemented as shader programs that are available on modern graphics hardware and lead to a shorter processing time.

In our render setup floating point cube maps are used as environment maps to represent the set of rays for the point of interest. Each texel represents a solid angle and stores the relative radiance value that is accumulated from the colors of all rays within the texel's solid angle. Each of the six cube map sides is handled as the image plane of a camera that is located at the point of interest. Depending on the cube map side, the camera is facing in the +X, -X, +Y, -Y, +Z or -Z direction.

All rays lying inside the texel's solid angle will be projected on the same texel. This is accomplished by simply rendering each Sliced Image with a three-pass shader into the cube map side.

The first pass evaluates for each pixel of the cube map side the denominator of equation 1 and the second pass evaluates the numerator. Additive blending is used to implement the sum of weighted color values. Results of the first and second pass are rendered into a temporary texture and the final pass calculates the quotient of both for each pixel.

As seen in fig. 13, the result is a cube map that contains all available environmental lighting for the point of interest as relative radiance values, although the cube map contains artifacts due to the very rough estimation of depth values.

(a) (b)

Figure 12: Estimating the depth map: (a) Original image with 3D key points. (b) The corresponding depth map as gray-scale image.

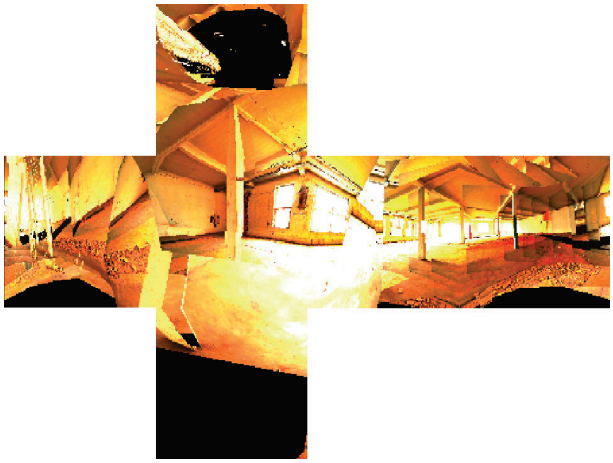

Figure 13: Rendered Cube Map describing the environmental lighting for the point of interest

Once the cube map has been rendered it can be used for various rendering techniques. For instance, it can be used to render specular surface by using reflection mapping. By applying a blur filter on the cube map contents, specular materials with different levels of shininess can be simulated.

In order to perform real-time capable illumination of 3D objects, the cube map's lighting information needs to be transformed to a simpler representation. As seen in fig. 14, our implementation extracts 16 directional light sources from the environment map using Debevec's approach[10]. These directional lights are used to illuminate the embedded virtual object. Additionally, the brightest light source is used to render the shadow of the virtual object.

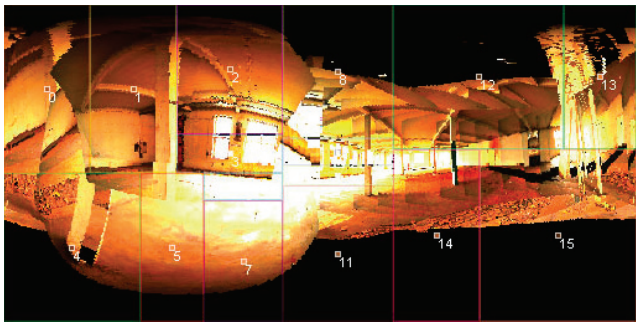

Figure 14: Cylinder projection of cube map with extracted directional lights

4.3 Run-time behavior

This subsection discusses the processing times for two exemplary 3D Photo Collections with 27 as well as 47 JPEG images. The format of the images is 24 bpp and their resolution is 2144x1424 pixel. A machine with an Intel Core i7 CPU 2.67 GHz, 6.00 GB RAM and two NVIDIA GeForce GTX 285 was used. The generation of the depth maps takes 397.2 seconds for the scene with 27 images and 676.579 seconds for the scene with 47 images. The rendering of sliced images is done in real-time. Due to its massive parallelization the generation of cube-maps takes not more than 0.358 seconds for the scene with 47 images. The run-time for the extraction of directional lights from the cube-map does not depend on the scene complexity, only on the resolution of the cube-map. In our setup

	Scene 1	Scene 2
Scene Complexity	27 images	47 images
Depth Map Generation	397.2 sec.	675.6 sec.
Cube Map Generation	0.188 sec.	0.358 sec.
Directional Light Extraction	0.321 sec.	0.323 sec.
Overall preprocessing time	397.709 sec.	676.281 sec.
Display of lit model and scene with occlusion handling	44.3 fps	35.1 fps

Table 1: Processing times of our occlusion handling and image-based lighting feature.

a cube-map with a resolution of $6 \times 128 \times 128$ pixels was used, which took less than 0.323 seconds (see table 1).

5 SUMMARY AND FUTURE WORK

5.1 Summary

A novel approach has been presented for dealing with occlusion problems and image-based lighting of embedded virtual objects in image-based 3D worlds. This approach relies on 3D Photo Collections that can be automatically constructed from unordered images with varying exposure.

Occlusion problems have been addressed by generating Sliced Images. These are created by segmenting the photos with a gradient-based contour tracing algorithm. The depth of these contour segments (called slices) are estimated by related 3D key points, which represents a sparse model of the spatial scene structure of the photos.

The generated Sliced Images are also used to accumulate the environmental lighting for a point of interest. The result is a HDR environment map. In our implementation, 16 directional light sources are extracted from the environment map and used to illuminate virtual objects.

5.2 Future Work

In future work, we will enhance our contour tracer for detecting polygons directly. This avoids potential loss of objects with less gradients.

The generation of the depth map does not contain potential merging of segments. In future work, all segments may be reviewed for potential consolidating. Therefore, several combinations of conditions are possible. Neighboring segments can be merged, when segments have

- similar depth and similar hue,
- similar depth and similar light intensity,
- similar depth, similar hue and similar saturation, or
- combinations of all described similarities.

Concluding from these observations, the merging is parameterizable. The larger the parameter spaces are, the more segments grow together. The more segments are merged, the less the (merged) segment depth value is exact.

In the current approach, the depth reconstruction of an image does not depend on other images. This means the advantage of having multiple views of the real scene is not exploited. This could be achieved by fitting small scale geometrical objects (e.g. spheres) into the point cloud. After this step they could be used to project the images of the scene on them. A gain in depth precision would also increase the quality of the presented image-based-lighting approach, since the origin of the light is modeled with higher accuracy.

In addition, we plan to implement stereoscopic rendering in our viewer. This requires the modify of our rendering routines of Sliced Images. Based on the 3D key points and the contour-based segmentation, which is done for the occlusion handling, we will add stereoscopic depth into all photos.

ACKNOWLEDGEMENTS

This work was funded by BMBF (Federal Ministry of Education and Research, project no.: 17N0909). Furthermore, special thanks to Ekkehard Beier (EasternGraphics GmbH) for scientifical support to this project. Also, we thank Bastian Birnbach, Tobias Bindel and Stephan Rothe (Erfurt University of Applied Sciences) for their technical support.

REFERENCES

[1] F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, 1991.

[2] R. T. Azuma. A Survey of Augmented Reality. In *Presence: Teleoperators and Virtual Environments 6*, 1997.

[3] Y. Bando, B.-Y. Chen, and T. Nishita. Extracting depth and matte using a color-filtered aperture. In *SIGGRAPH Asia '08 papers*, pages 1–9, 2008.

[4] R. Brinkmann. *The Art and Science of Digital Compositing*. Morgan Kaufmann, 1999.

[5] J. Canny. A computational approach to edge detection. *Readings in computer vision*, vol.184, 1987.

[6] B. Chanda and D. D. Majumder. Boundary-based Description. In *Digital Image Processing and Analysis*, pages 312–314. Prentice-Hall of India Pvt.Ltd, 2004.

[7] B. Chanda and D. D. Majumder. Edge and Line Detection. In *Digital Image Processing and Analysis*, pages 239–277. Prentice-Hall of India Pvt.Ltd, 2004.

[8] N. Dachuri, S. M. Kim, and K. H. Lee. Estimation of few light sources from environment maps for fast realistic rendering. In *ICAT '05 Proceedings*, pages 265–266, 2005.

[9] P. Debevec. Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography. In *SIGGRAPH98*, pages 189–198. ACM, 1998.

[10] P. Debevec. A Median Cut Algorithm for Light Probe Sampling. In *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*, 2005.

[11] A. Fournier, A. S. Gunawan, and C. Romanzin. Common Illumination between Real and Computer Generated Scenes. Technical report, University of British Columbia, Vancouver, BC, Canada, Canada, 1992.

[12] Y. Furukawa and J. Ponce. Carved visual hulls for high-accuracy image-based modeling. In *SIGGRAPH '05 Proceedings*, page 146. ACM, 2005.

[13] S. Gibson, J. Cook, T. Howard, and R. J. Hubbold. Rapid Shadow Generation in Real-World Lighting Environments. In *Rendering Techniques*, pages 219–229, 2003.

[14] M. Goesele, J. Ackermann, S. Fuhrmann, C. Haubold, R. Klowsky, and T. Darmstadt. Ambient point clouds for view interpolation. *ACM Transactions on Graphics (TOG)*, 29(4):1–6, 2010.

[15] Google. Street View. `http://maps.google.com/`, 2007.

[16] T. Grosch. PanoAR: Interactive augmentation of omnidirectional images with consistent lighting. *Mirage 2005, Computer Vision / Computer Graphics Collaboration Techniques and Application*, pages 25–34, 2005.

[17] K. Kölzer, F. Nagl, B. Birnbach, and P. Grimm. Rendering Virtual Objects with High Dynamic Range Lighting Extracted Automatically from Unordered Photo Collections. In *ISVC 2009 Proceedings: Part II*, pages 992–1001, 2009.

[18] M. I. Lourakis and A. A. Argyros. The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the

Levenberg-Marquardt Algorithm. Technical report, Heraklion, Crete, Greece, 2004.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[20] Microsoft. Photosynth. `http://photosynth.net/`, 2008.

[21] F. Nagl, M. Friedl, A. Schäfer, and A. Tschentscher. Shader-Based-Image-Processor. In *GI Seminars 9. Informatiktage 2010*, pages 223–226, 2010.

[22] F. Nagl, P. Grimm, and D. Abawi. IP3D - A Component-based Architecture for Image-based 3D Applications. In *SEARIS@IEEEVR2010 Proceedings, IEEE VR 2010 Workshop*, pages 47–52, 2010.

[23] F. Nagl, P. Grimm, B. Birnbach, and D. Abawi. PoP-EYE environment: Mixed Reality using 3D Photo Collections. In *ISMAR 2010 Poster Proceedings*, pages 255–256, 2010.

[24] F. Nagl, K. Kölzer, P. Grimm, T. Bindel, and S. Rothe. Congrap – contour detection based on gradient map of images. In G. S. Hamid R. Arabnia, Leonidas Deligiannidis, editor, *Proceedings of the 2011 International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV 2011)*, volume II, pages 870–875. CSREA Press, USA, 2010.

[25] H. Regenbrecht. *Faktoren für Präsenz in virtueller Architektur*. PhD thesis, Bauhaus-Universität Weimar, 2000.

[26] M. Robertson, S. Borman, and R. Stevenson. Estimation-theoretic approach to dynamic range enhancement using multiple exposures. *Journal of Electronic Imaging*, 12:219, 2003.

[27] N. Snavely. Bundler: Structure from Motion for Unordered Image Collections. `http://phototour.cs.washington.edu/bundler/`, 2010.

[28] N. Snavely. Bundler v0.4 User's Manual. `http://phototour.cs.washington.edu/bundler/bundler-v0.4-manual.html`, 2011.

[29] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. In *SIGGRAPH '06 Proceedings*, pages 835–846, 2006.

[30] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the World from Internet Photo Collections. *Int. J. Comput. Vision*, 80(2):189–210, 2008.

[31] J. Zhu and Z. Pan. Occlusion registration in video-based augmented reality. In *VRCAI '08 Proceedings*, pages 1–6, 2008.