Interactive Augmented Omnidirectional Video with Realistic Lighting

Nick Michiels, Lode Jorissen, Jeroen Put, and Philippe Bekaert

Hasselt University - tUL - iMinds, Expertise Centre for Digital Media
Wetenschapspark 2, 3590 Diepenbeek, Belgium
{nick.michiels,lode.jorissen,jeroen.put,philippe.bekaert}@uhasselt.be

Abstract. This paper presents the augmentation of immersive omnidirectional video with realistically lit objects. Recent years have known a proliferation of real-time capturing and rendering methods of omnidirectional video. Together with these technologies, rendering devices such as Oculus Rift have increased the immersive experience of users. We demonstrate the use of structure from motion on omnidirectional video to reconstruct the trajectory of the camera. The position of the car is then linked to an appropriate 360° environment map. State-of-the-art augmented reality applications have often lacked realistic appearance and lighting. Our system is capable of evaluating the rendering equation in real-time, by using the captured omnidirectional video as a lighting environment. We demonstrate an application in which a computer generated vehicle can be controlled through an urban environment.

Keywords: omnidirectional video, realistic lighting, product integral rendering, structure from motion

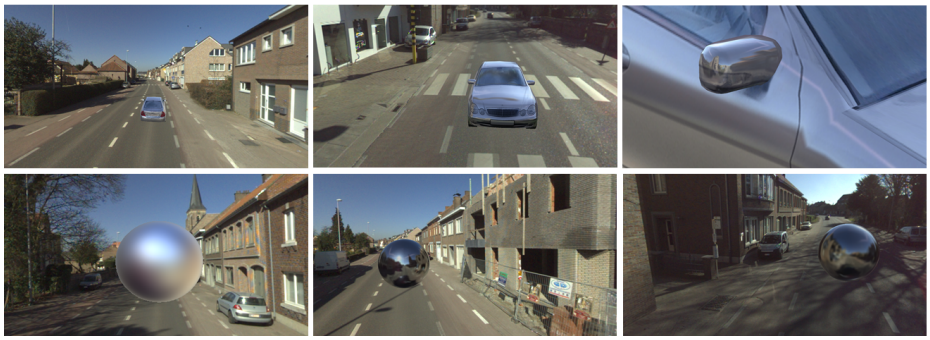


Fig. 1. Illustration of real-time augmented rendering of virtual objects in an omnidirectional environment. The 360° video is used as environment map for realistic lighting.

1 Introduction

Omnidirectional video is an emerging medium that gives viewers a 360° panoramic experience. Previous work focused on the real-time capturing and rendering of omnidirectional video on a 360° projection screen [24, 5], in a head-mounted display like Oculus Rift [29] or other applications like Illumniroom [14]. We demonstrate the use of structure from motion on omnidirectional video to reconstruct the trajectory of the camera. The structure is used to accurately track the camera poses in a large environment. Extending this omnidirectional content with camera position tracking creates a global coordinate system in which virtual objects can be augmented. This makes it possible to bridge the gap between video and computer graphics. Possible applications are interactive worlds, architectural modelling and marketing.

Up until now, most augmented reality applications have often lacked realistic lighting [13]. This paper shows how to leverage tracking information and use that to feed the correct environment frame to a realistic renderer. The existing environment lighting from the captured 360° video can be used to augment realistic objects in the scene in real-time. For this, the augmented virtual objects need to be synchronized with the lighting information of its position. Our goal is to render virtual objects with lighting that is consistent with the captured video, so that the objects are seamlessly integrated into the environment. An example of a technique is to find the main light source of the scene using the shadows in the original image [2]. This way, realistic shadowing can be achieved, but all high frequency lighting information is lost. Since we are working with an omnidirectional camera, we have lighting information from all directions for all camera poses. This paper explores how that lighting information can be used to feed to the rendering. As explained before, tho achieve this, a proper renderer that is able to render distant view-dependent high frequency lighting and detailed spatial and angular reflection is required. Our proposed system is capable of evaluating the rendering equation for distant light in real-time, based on precomputed radiance transfer using spherical radial basis functions. Our system is demonstrated with an application in which an augmented vehicle can be controlled through an urban environment (Figure 1).

The paper is organized as follows. The first section describes the related work. Then, we present the three steps necessary to achieve our goal of a real-time augmented renderer. Section 3.1 explains how to capture, process and render the omnidirectional data with a custom camera. Section 3.2 describes how to extract the camera poses of the omnidirectional camera using structure from motion. Section 3.3 gives a solution for a real-time renderer that uses the tracked frames to realistically render lighting and reflectance effects. The results of this work and an application are shown in Section 4. Finally, in Section 5, we conclude and discuss the future work of this paper.

2 Previous Work

Debevec [4] was one of the first to propose image-based lighting (IBL), where the capturing of real-world lighting is used to render virtual objects. The environment lighting is described as a single light probe image. A nice overview of photorealistic and non-photorealistic augmented reality applications is made by Haller et al. [13]. They

compare both approaches using ARToolKit markers to track the camera its pose. In the photorealistic approach, they show how to use shadow volumes and bump mapping.

Agusanto et al. [1] show an application where image-based rendering techniques are used to incorporate virtual objects in augmented reality content using environment illumination maps. They do not capture any environment map, but use existing single shot light probes. The disadvantage is that they are not able to render with interactive and dynamic lighting.

Other techniques [16, 2] focus on shadow effects in augmented reality applications. They identify the most important light sources in the light probe and use them to cast shadows of the augmented objects so that they are consistent with the real world. In terms of other lighting and reflectance effects, they lack visual realism.

Grosch et al. [11] showed how to make the light of a room consistent with the rendered virtual objects by reconstructing the geometry of a single light probe. The reconstruction makes it possible to add new lights sources to the light probe as well as place new objects in the room and project the light probe for different positions in the room. They mainly focus on consistency of the light conditions in a room, but not on how to realistically render objects with these new light conditions. Additionally they only use one light probe for the entire scene, which makes their algorithm not applicable to larger scenes. Later, Grosch et al. [12] studied how to use irradiance volumes to place virtual objects in a real-life Cornell Box. The external illumination is captured with a HDR camera and a fisheye lens. Direct and indirect lighting effects are simulated for the virtual objects. The rendering is done at interactive frame rates, but the geometry of the objects cannot change easily which makes the rendering of dynamic scenes not possible.

Papagiannakis et al. [23] and Gierlinger et al. [9] have integrated the full precomputed radiance transfer (PRT) [27, 21, 22] in a mixed reality application using HDR input images. They use spherical harmonics to precompute the three factors of the radiance transfer. They are able to render high quality and realistic lighting in real-time but require a great amount of precomputation. Since precomputation of visibility is necessary, the results are limited to static scenes. Additionally, the use of spherical harmonics are suboptimal for high frequency lighting conditions.

This paper will also use precomputed radiance transfer to render the scene with accurate environment lighting. To make sure that the high frequency light conditions are preserved, the three factors of the radiance transfer are represented with spherical radial basis functions (SRBFs) instead of spherical harmonics [30]. Furthermore, using this spherical radial basis representation, the amount of precomputation will decrease drastically: the bidirectional reflectance distribution function (BRDF) is approximated with few SRBFs; the visibility can be traced using cone tracing where the cones are approximated with SRBFs; and the environment frame can be efficiently transformed into a SRBF representation using an hierarchical approximation algorithm. Since almost no precomputation is required, the three factors can change at any time making the rendering system very flexible and interactive. This is particularly important since our application of driving a virtual vehicle in an urban environment requires a frequent change in all three factors.

(a)

(b)

(c)

Fig. 2. Capturing of omnidirectional video. (a) Our custom designed omnidirectional camera consists of 6 cameras, all capturing at 25 fps with a resolution of 1600×1200. (b) The omnidirectional camera mounted on a vehicle to capture the streets of an urban environment. (c) The full panoramic stitched frame of all six cameras, which can be done in real-time.

3 Our Approach

The goal of this paper is to augment omnidirectonal video (ODV) with new virtual objects that are realistically lit by their environment. The user must be able to interact with the video by moving around the virtual object, while at the same time the lighting and reflectance conditions stay consistent with the captured environment. This requires a system with the following three stages:

1. Stitching and rendering of ODV
2. Offline reconstruction the camera trajectory of ODV using structure from motion
3. Real-time realistic rendering of virtual objects using the ODV for lighting

These three stages will now be explained in more detail.

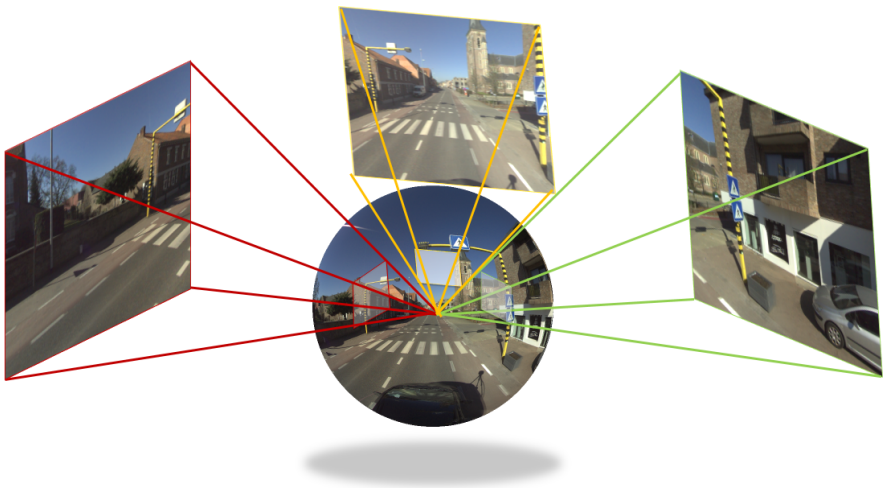


Fig. 3. Rendering of omnidirectional content. A virtual camera (red, yellow or green) can be placed in the spherical warped ODV frames. This gives the viewer a full 360° interactive experience, since one can freely look around with the camera.

3.1 ODV capturing and rendering

Capturing is performed with a custom designed camera, built with six cameras at 60° intervals with approximately 50% coverage between adjacent cameras. The omnidirectional camera is depicted in Figure 2(a). Each camera has a resolution of 1600×1200. All six cameras are synchronized and capture at 25 frames per second. Next, we mounted the camera on top of a vehicle and drove through a local city, which is demonstrated in Figure 2(b). The stitching and rendering of the different camera frames are performed in real-time on the graphics card. Figure 2(c) depicts an example of such stitched frame in an equirectangular representation. The rendering of such frames requires the warping of the equirectangular stitched frame onto a sphere where the virtual camera is positioned within the sphere. This way, the camera can be rotated in any direction, so that the user can freely experience a 360° walkthrough in the video environment, as illustrated in Figure 3. This step is performed in real-time, but for this application it is not required.

3.2 Camera tracking using structure from motion

The ability of adding new virtual objects in the captured video of the previous section requires the pose estimation of the omnidirectional camera throughout the sequence. This step is of significant importance and aims to determine the alignment between synthetic objects and the real world in order to render them at the correct position. Once the alignment is complete, an appropriate omnidirectional video frame can be identified for each virtual object which will serve as environment lighting so that realistic lighting can be achieved. This section will explain how to adapt a structure from motion algorithm to

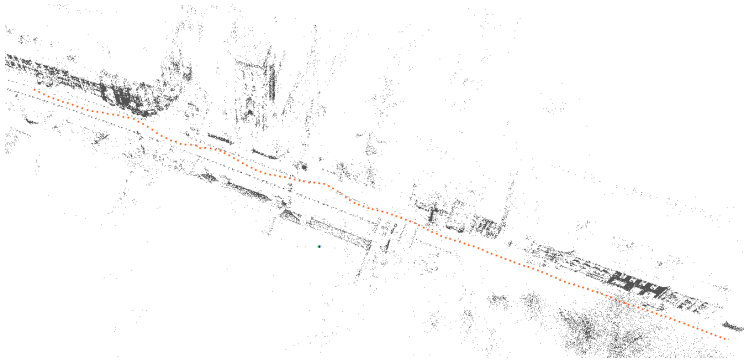


Fig. 4. Initial reconstruction of the camera poses using visual odometry without sparse bundle adjustment. The accuracy of the poses is very poor.

make it work with omnidirectional content. The result is accurate tracking information of the omnidirectional camera for a large environment.

The first step of a structure from motion algorithm is feature tracking. Since the stitched omnidirectional video frames have still some ghosting artifacts which will possibly degrade the tracking accuracy, feature tracking was done on the individual camera images (the frames from the cameras that make up the omnidirectional camera). Since the poses of the individual cameras relative to the center of the ODV camera are already known (this information is extracted during ODV camera calibration), we do not need to estimate the pose of each individual camera for each frame. Instead, the pose of the complete ODV camera is estimated using only the image information of the individual cameras, resulting in a more accurate pose estimation.

As explained earlier the algorithm works on the individual cameras of the ODV camera. The first step in this process is to remove the distortion from the images. In our current implementation we apply a Shi-Tomasi corner detector on the undistorted input images [26]. Only features of good quality are stored. The features are then tracked to the next frame of the corresponding camera using Lucas-Kanade optical flow tracking [20]. We chose the Lucas-Kanade tracker instead of a descriptor based matching method because we want the number of frames in which a feature is tracked to be as high as possible. Longer feature tracks result in more shared information between the poses of the ODV camera and thus result in a more stable pose estimation. In case of short feature tracks there is less shared information and the estimated poses often show more jitter when rendering synthetic objects. The optical flow algorithm can however result in erroneous matches. Therefore, we use the trifocal tensor constraint to filter out incorrect matches. The trifocal tensor is calculated for three adjacent ODV frames with a RANSAC approach [7]. The input of the RANSAC step consists of the features that are visible in all three frames. A subset of these features is used to determine a trifocal tensor, and the trifocal tensor is used to check whether a feature is an inlier or not (the position in the third frame needs to be close to the estimated position).

For each feature a SIFT descriptor [19] is extracted, necessary to match the features between the cameras. Feature matching between the cameras allows us to connect

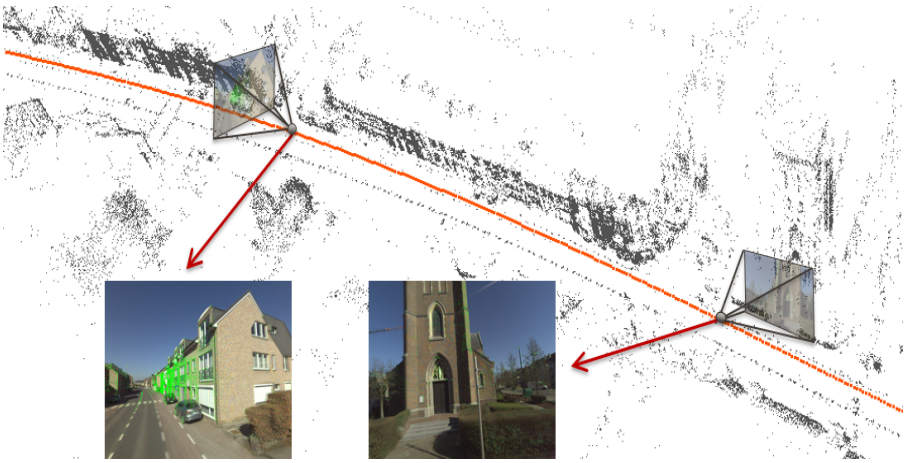


Fig. 5. Structure from motion is used to reconstruct the trajectory of the road (depicted in orange). The trajectory is used to align the rendered objects with the real-world scene.

shorter feature tracks into longer tracks. This is especially useful when a feature leaves one camera and shows up in another one. A cross-check is applied to make sure that the features match in both directions. The remaining features are checked against the epipolar constraint.

Estimating the trifocal tensor for outlier removal also has the advantage that one can extract the fundamental matrices (from camera 1 to 2 and from camera 1 to 3). These encode the relative position and orientation of the cameras and thus can be used to determine an estimate of the poses of the ODV camera. This process is similar to visual odometry (VO) [25, 8] and, as is to be expected with VO, the resulting positions are far from optimal since the pose of the camera is only estimated in relation to the previous pose. The initial estimation of poses is shown in Figure 4. A better option is to globally estimate the pose using the information gathered in all ODV frames. This is done by applying bundle adjustment.

Bundle adjustment simultaneously updates the poses of the cameras and the 3D world positions to minimize the mean error between the reprojected world points and their corresponding features. The error metric is defined as a function of the distance between the position of the feature and the reprojection. The sparse bundle adjustment implementation of Lourakis and Argyros [18] is used. As input, the algorithm requires initial estimates of the poses and the world positions of the features. These initial world positions are obtained by doing a SVD based triangulation, followed by an optimization step that also reduces the reprojection error

The results of bundle adjustment are refined ODV camera poses and a sparse 3D structure that aligns to objects in the world. Figure 5 shows the poses and the structure of the track after bundle adjustment as well as examples of the structure rendered on top of the omnidirectional video using an estimated pose. One can clearly see that the 3D points are aligned with their corresponding features in the video and that the estimated

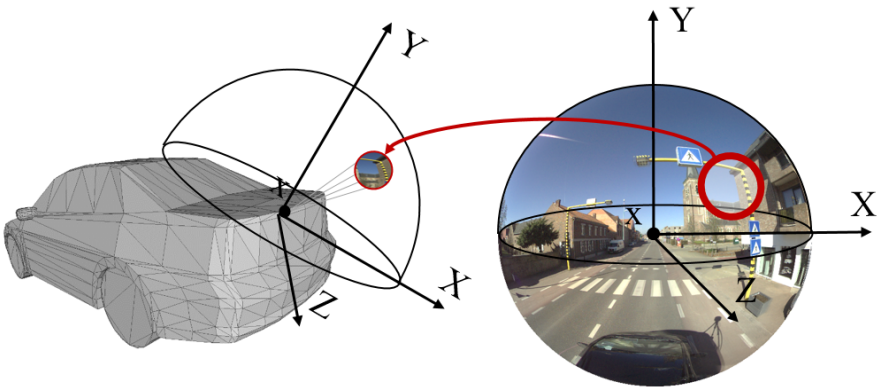


Fig. 6. Rotation of a SRBF of an environment map defined in the global frame (right) to a SRBF of an environment map defined in the local frame of a vertex is done by simply rotating its center.

positions of the camera better represent the true positions of the camera compared to the initial poses of visual odometry in Figure 4.

3.3 Real-Time Augmented Rendering

High quality and realistic rendering of objects will drastically increase the immersive experience of augmented reality applications. To achieve this, the rendering requires detailed natural lighting and realistic reflectance of materials. View-dependent effects like a change in eye direction across surfaces will improve visual realism. This dynamically varying viewpoint will require a reflectance representation defined in 6D (light direction, view direction and surface position). Since detailed variations in reflectance will visually impact the end result, the 6D functions needs to be evaluated densely. Naive sampling of the rendering equation [15] at every point in the scene will result in a correct calculation of the radiance reaching the observer, but will also be very intractable. Solutions using precomputed radiance transfer (PRT) [27, 21, 22] are able to render efficiently under the above mentioned conditions. Here the light transport is precalculated in a factored form. The rendering equation is then evaluated at each point x, with view direction ω_0 by calculating the triple product integral over the hemisphere of the factored representation:

$$B(x,\omega_o) = \sum_i \sum_j \sum_k V_i \; \rho_j(\omega_o) \; L_k \int_\Omega \Psi_i(\omega) \; \Psi_j(\omega) \; \Psi_k(\omega) d\omega \qquad (1)$$

with visibility V, reflectance ρ, environment lighting L.

Previous methods used Haar wavelets as a representation for the three factors of the triple product rendering equation [22]. An important disadvantage of wavelet-based methods is the absence of an efficient rotation operator in the wavelet domain, which is required to rotate the environment in the local frame for each vertex. Additionally, a per-vertex 2D slice of the bidirectional reflectance distribution function (BRDF) needs

to be sampled at execution time. We use spherical radial basis functions (SRBF) to represent each element of the rendering equation [30]. A SRBF is a rotation-invariant function depending on the parameter v describing a direction on the sphere S^2:

$$G(v; p, \lambda, \mu) = \mu e^{-\lambda} e^{\lambda(v \cdot p)} \tag{2}$$

where $P \in S^2$ is the center of the SRBF, $\lambda \in (0, +\infty)$ is the lobe sharpness and $\mu \in \mathbb{R}$ is the lobe amplitude. This spherical radial basis function can now be used to approximate any spherical function to an arbitrary degree of accuracy by mixing various SRBFs with different lobe sizes and positions:

$$F(v) = \sum_{l-1}^{n} G(v; p, \lambda_l, \mu_l) \tag{3}$$

Unlike Haar wavelets, spherical functions represented with SRBFs can easily be rotated on the sphere by simply rotating its SRBF lobe centers (depicted in Figure 6). Furthermore, as shown by Tsai et al. [28], the product and convolution of Gaussians can be evaluated analytically. By using these characteristics, the convolution of three Gaussians results in a scalar, making the triple product integral calculations of the rendering equation straightforward:

$$\begin{aligned} &\int_{S^2} G_1(v; p_1, \lambda_1, \mu_1) G_2(v; p_2, \lambda_2, \mu_2) G_3(v; p_3, \lambda_3, \mu_3) dv \\ &\quad = \frac{4\pi\mu_1\mu_2}{e^{\lambda_1+\lambda_2+\lambda_3}} \frac{\sinh(\|r\|)}{\|r\|} \end{aligned} \tag{4}$$

where $r = \|\lambda_1 p_1 + \lambda_2 p_2 + \lambda_3 p_3\|$. Once all three factors of the rendering equation are represented in the SRBF basis as described above, efficient rendering of a realistically lit virtual object is possible and the object can be integrated in the omnidirectional scene in real-time. The next sections show how to obtain the basis transformations for lighting, reflectance and visibility.

Environment Map Using the tracking information of Section 3.2, we can map the reconstructed 3D positions to an appropriate omnidirectional video frame. These omnidirectional video frames can then be used as environment lighting for virtual objects

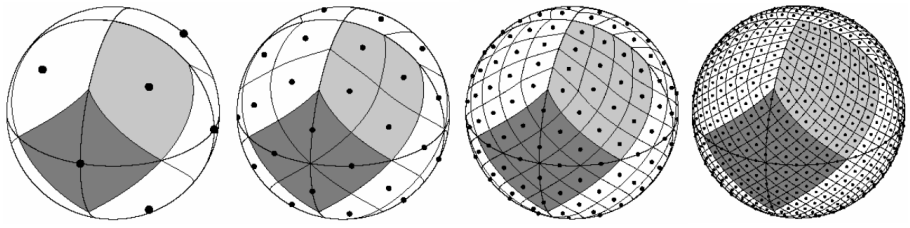


Fig. 7. Healpix distribution scheme subdivides the sphere in equal area parts. The Healpix quadrilaterals of a specific level can be subdivided in 4 new equally-sized Healpix quadrilaterals on the next level [10].

for that specific position. The environment map frames are approximated with a multi-level hierarchical SRBF model. The domain of the spherical function is subdivided in different hierarchical layers of SRBFs with increasing density and decreasing lobe size. The SRBF centers of a specific level are uniformly distributed over the spherical surface. This paper uses the Healpix [10, 17] distribution on the sphere. Healpix is optimal for distributing the spherical surface in equal area parts (see Figure 7). Moreover, it is suitable for hierarchical algorithms, since one Healpix quadrilateral on a specific level can be easily subdivided in 4 new equally-sized Healpix quadrilaterals on the next level. Transforming a spherical signal into a SRBF representation is started at the root level of the multiscale hierarchy, consisting of only 12 Healpix SRBF centers. Only SRBFs that decrease the residual error of the signal will be inserted. Once all necessary SRBFs are inserted, the full residue can be calculated. This residue will serve as the input signal for the next level (48 Healpix SRBF centers). The process is iterated until the maximum level is reached or the residue error is lower than a specified threshold [6]. This transformation process avoids SRBF estimation procedures which often use costly optimization algorithms. It is therefore fast and can happen for real-time video.

BRDF Instead of sampling the 2D slices of the BRDFs in the pixel domain, these can now be directly approximated with few SRBFs. The BRDF models used in this paper all are based on microfacet theory, meaning that the specular lobe ρ_s can be described in terms of a normal distribution function (NDF) $D(h)$, with h the halfway vector and a remaining factor $M_o(i)$:

$$\rho_s(o,i) = M_o(i)D(h) \quad h = \frac{o+i}{\|o+i\|} \tag{5}$$

BRDFs that are expressed in terms of the normal distribution (NDF) can be approximated with spherical Gaussians. $D(h)$ can be approximated using a single spherical Gaussian for isotropic models or multiple spherical Gaussians for anisotropic models. $M_o(i)$ is very smooth and can be approximated by a constant. We implemented multiple BRDF models. For example, fitting the Blinn-Phong model with a Gaussian lobe is done as follows:

$$\begin{aligned} M_o(i) &= \frac{n+2}{2\pi} \\ D(h) &= (h \cdot n)^{\lambda} \approx G(n; h, \lambda, 1) \end{aligned} \tag{6}$$

To obtain the 2D BRDF slice $\rho_s(o,i)$ for a specific view direction o, we first need to warp the lobe described in terms of the halfway vector into the lobe defined in terms of the view direction. The warp for every lobe is specified as:

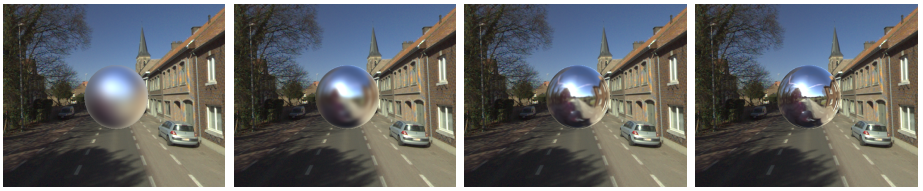
$$\begin{aligned} p_i^W &= 2(o \cdot h)h - o \\ \lambda_i^W &= \frac{\lambda_i^D}{\tau(p_i^D)} \\ \mu_i^W &= \mu_i^D \end{aligned} \tag{7}$$

where the differential area of the warp $\tau(h) = 4\|h \cdot o\|$ is defined as the determinant of the Jacobian of the warp function. The details of fitting different BRDF models and warping them on the sphere are described in detail by Wang et al [30].

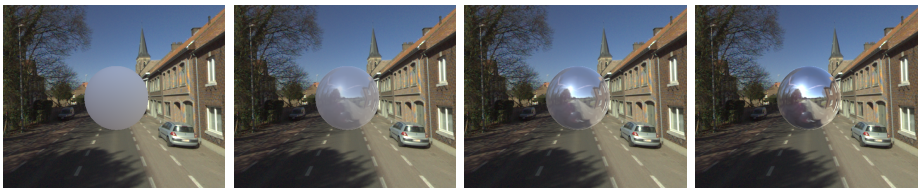
Visibility The visibility can be traced in real-time using voxel cone tracing [3]. Voxel cone tracing requires virtual objects to be rasterized using a real-time voxelization algorithm. This is done by rasterizing the geometry along the three main axes of the scene. The viewport resolution defines the resolution of the 3D voxel volume. We have observed that a 256^3 voxel volume yields plausible visibility. The voxel volumes are stored in OpenGL 3D textures in contrast with the sparse octree representation of Crassin et al. [3]. The main advantage of using 3D textures is that they require no octree traversal, making them ideal for parallelism on GPU. In the future, sparse 3D textures will become available, allowing the algorithm to work with bigger objects. A next step is to generate MIP-maps for the 3D texture. These MIP-mapped versions of the volume will be extensively used in the cone tracing for LOD, since the footprint of cones will increase with the distance to the center, allowing lower resolution MIP-maps to be used. We use importance sampling in order to know where to trace cones for visibility. For every product of a lighting SRBF and reflectance SRBF, the support can be determined and used to sample the directions for cone tracing. The cone can then be seen as a SRBF on its own, and can be applied in the triple product integral.

4 Results

Our real-time augmented renderer is implemented on a system with an Intel Xeon™ dual six core processor and a nVidia GeForce GTX 780 graphics card. The stitching



(a) Phong ($\alpha = 10$) (b) Phong ($\alpha = 100$) (c) Phong ($\alpha = 1000$) (d) Phong ($\alpha = 10000$)



(e) kd = 1.00 (f) kd = 0.75 (g) kd = 0.5 (h) kd = 0.25

Fig. 8. Rendering with different material properties. Top row: Phong BRDF whith an exponent α ranging from 10 to 10000. Bottom row: mixing of materials with a kd percentage of a Lambertian BRDF added to a $(1 - kd)$ percentage of a Phong ($\alpha = 10000$) BRDF.

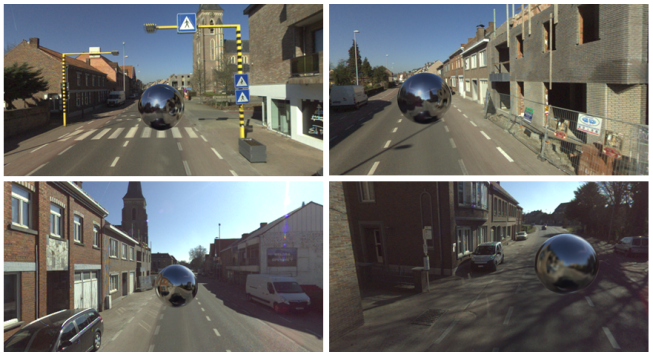


Fig. 9. Rendering of a sphere. Different positions of the sphere will result in different lighting conditions.

and rendering of the omnidirectional video is performed on the GPU in GLSL shaders. We created a viewer where the user can walk through the video and can manipulate the viewing direction of the camera. This viewer can be outputted to a normal screen or to more immersive hardware like Oculus Rift or a omnidirectional cave.

Structure from motion is done offline as a preprocessing step. The focus of this step lies on quality and accuracy and not so much on time performance since the application requires only the reconstructed trajectory of a video that is not captured at runtime.

The transformation of the omnidirectional video frames to spherical radial basis functions is done on the CPU and later passed to the GPU. In GLSL shaders, the different models of the BRDF and the spherical warp are implemented and together with the SRBFs of the environment lighting and possibly the visibility, they are used to evaluate the triple product integral of Section 3.3. At this time, we have not integrated the visibility factor into the renderer. Our cone tracing is not optimized to work in real-time yet, but, as explained in Section 3.3, once the cone tracing is fully operational, the integration is straightforward. For basic ambient occlusion, the cone tracing is working, since only few cones are traced over the hemisphere. An example of ambient occlusion of a car using cone tracing is depicted in Figure 11(a). When we render the same car with our real-time renderer and scale with this ambient occlusion factor we get Figure 11(b). In the future, more detailed geometry of the SfM from Section 3.2 can be used to integrate occlusions from the environment. A main advantage of using SRBFs as a representation is that BRDFs can easily be approximated with few SRBFs which can be reconstructed at real-time using only the parameters of the BRDF model. For each point in the virtual object, the system only needs to know its BRDF parameters and then an appropriate BRDF radial basis function can be assigned. We have support for the Lambertian, Phong, Blinn-Phong, Ward and Cook-Torrance BRDF models. In Figure 8, we show a sphere model inside our captured environment, all rendered with different material parameters. The top row of the Figure shows the Phong BRDF with a exponent parameter ranging from 10 to 10000. The bottom row shows the mixing of a specular Phong BRDF with a Lambertian BRDF. Now the percentage of mixing the diffuse Lambertian is ranging from 0.25 to 1.00. Figure 10 depicts the results of a more

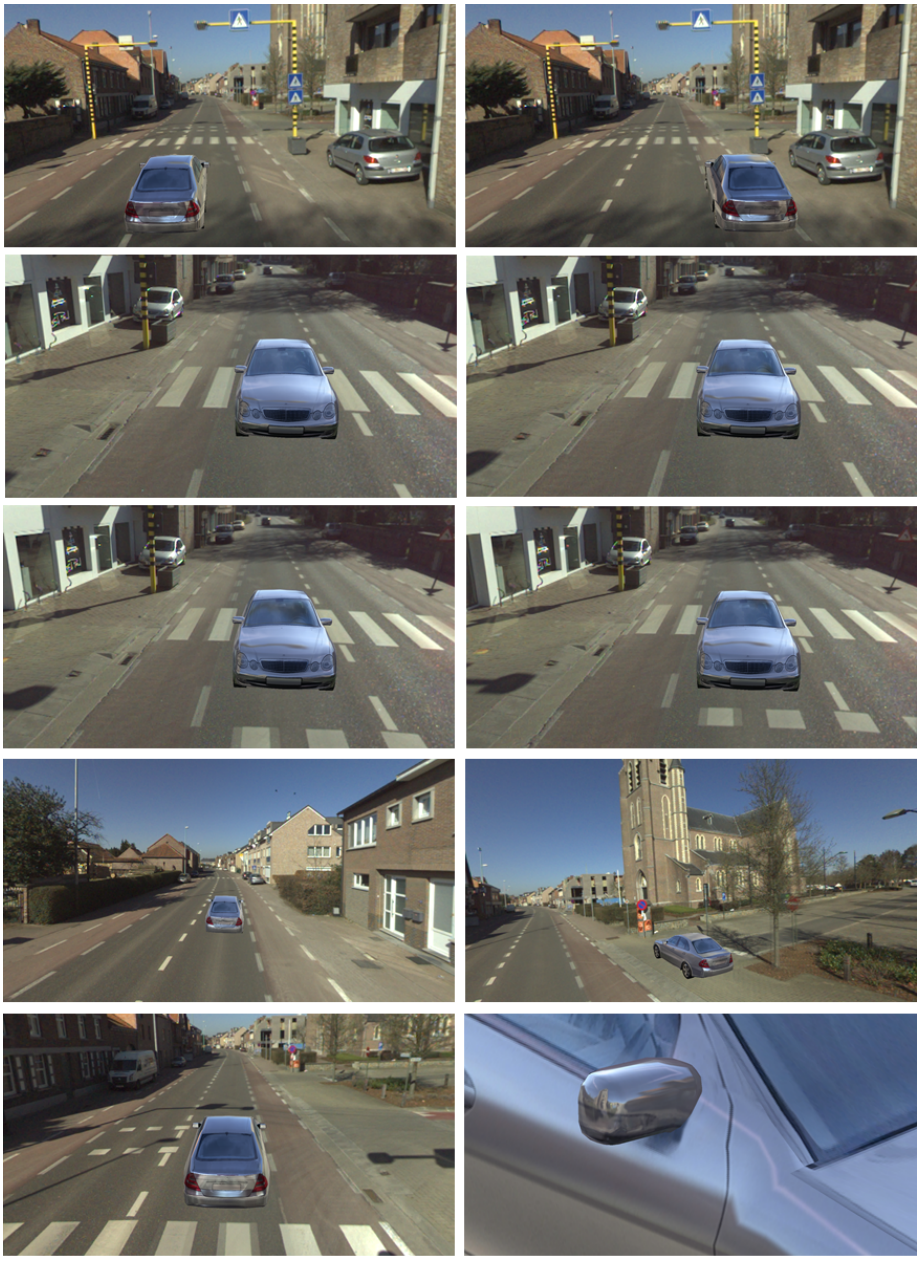
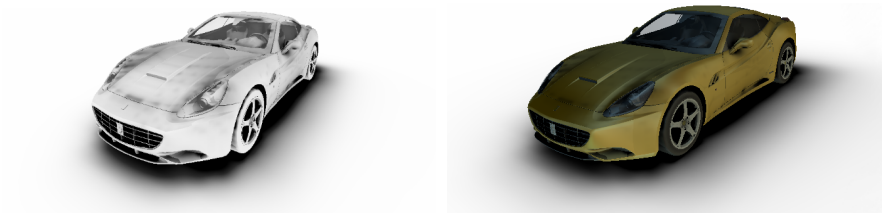


Fig. 10. Rendering of a user-controlled vehicle in an urban environment. Row 1 shows the change in reflection when the user is horizontally moving the vehicle. Row 2 and 3 depict the change in reflection when driving the car forward. Row 4 and 5 show the car from different angles.



(a) (b)

Fig. 11. Rendering of (a) ambient occlusion using cone tracing and (b) environment map, BRDF and ambient occlusion.

realistic model, i.e. a vehicle. A model of a Mercedes with texture mapping is inserted into the scene. Furthermore, we assigned a realistic material with a mixture of BRDFs to the model. The figure clearly shows accurate view depended lighting and detailed spatial and angular reflections when the car is moved in the environment. We provide a video of our application[1]. In Figure 9 we show the same sphere and assigned a high specular material to make sure all lighting effects of the environment onto the sphere can be seen. We chose a Phong BRDF with a big exponent factor ($N = 10000$). It is now possible to perceive a change in reflection when the sphere is moved around in the urban environment.

5 Conclusion

This paper presented an approach for augmenting omnidirectional video with interactive virtual objects rendered with realistic lighting and reflectance properties. The method used offline feature tracking to align positions of virtual objects with positions of the real-world. We proposed to use the omnidirectional video frames as environment lighting for the virtual objects and showed how to use the tracking information to synchronize the correct environment map with the virtual objects. Furthermore we explained how to transform the environment lighting, BRDF properties and possibly the visibility to spherical radial basis functions in order to render the virtual objects in real-time and with realistic reflectance and lighting integrated over the hemisphere. Currently, visibility is not entirely integrated in the rendering, but we showed how to use cone tracing to solve this issue. Future work is to integrate our cone trace approach in the renderer. Further development of the structure from motion step can result in more dense geometry. This geometry can be used to cast shadows onto the objects as well as correct occlusions. For now, the renderer is limited to distant light. But near-field lighting can be sampled, similar to the cone tracing algorithm of the visibility factor.

Acknowledgements This work has been made possible with the help of a PhD specialization bursary from the IWT and FWO. The authors acknowledge financial support from the European Commission (FP7 IP SCENE).

[1] `http://www.youtube.com/watch?v=gwftfqfY9WM`

References

1. Agusanto, K., Li, L., Chuangui, Z., Sing, N.W.: Photorealistic rendering for augmented reality using environment illumination. In: Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on. pp. 208–216 (Oct 2003)
2. Arief, I., McCallum, S., Hardeberg, J.Y.: Realtime estimation of illumination direction for augmented reality on mobile devices. In: Color and Imaging Conference. pp. 111–116. IS&T and SID, Los Angeles, CA, USA (Nov 2012)
3. Crassin, C., Neyret, F., Sainz, M., Green, S., Eisemann, E.: Interactive indirect illumination using voxel-based cone tracing: An insight. In: ACM SIGGRAPH 2011 Talks. pp. 20:1–20:1. SIGGRAPH '11, ACM, New York, NY, USA (2011), http://doi.acm.org/10.1145/2037826.2037853
4. Debevec, P.: Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. pp. 189–198. SIGGRAPH '98, ACM, New York, NY, USA (1998)
5. Dumont, M., Rogmans, S., Maesen, S., Frederix, K., Taelman, J., Bekaert, P.: A spatial immersive office environment for computer-supported collaborative work - moving towards the office of the future. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. pp. 212–216. SIGMAP 2011 (2011)
6. Ferrari, S., Maggioni, M., Borghese, N.: Multiscale approximation with hierarchical radial basis functions networks. Neural Networks, IEEE Transactions on 15(1), 178–188 (Jan 2004)
7. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
8. Fraundorfer, F., Scaramuzza, D.: Visual odometry : Part ii: Matching, robustness, optimization, and applications. Robotics Automation Magazine, IEEE 19(2), 78–90 (June 2012)
9. Gierlinger, T., Danch, D., Stork, A.: Rendering techniques for mixed reality. Journal of Real-Time Image Processing 5(2), 109–120 (2010)
10. Gorski, K., Hivon, E., Banday, A., Wandelt, B., Hansen, F., et al.: HEALPix - A Framework for high resolution discretization, and fast analysis of data distributed on the sphere. Astrophys.J. 622, 759–771 (2005)
11. Grosch, T.: PanoAR: Interactive augmentation of omni-directional images with consistent lighting. In: Mirage 2005, Computer Vision / Computer Graphics Collaboration Techniques and Applications. pp. 25–34 (2005)
12. Grosch, T., Eble, T., Mueller, S.: Consistent interactive augmentation of live camera images with correct near-field illumination. In: Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology. pp. 125–132. VRST '07, ACM, New York, NY, USA (2007)
13. Haller, M.: Photorealism or/and non-photorealism in augmented reality. In: Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry. pp. 189–196. VRCAI '04, ACM, New York, NY, USA (2004), http://doi.acm.org/10.1145/1044588.1044627
14. Jones, B.R., Benko, H., Ofek, E., Wilson, A.D.: Illumiroom: Peripheral projected illusions for interactive experiences. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 869–878. CHI '13, ACM, New York, NY, USA (2013), http://doi.acm.org/10.1145/2470654.2466112
15. Kajiya, J.T.: The rendering equation. In: Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques. pp. 143–150. SIGGRAPH '86, ACM, New York, NY, USA (1986)

16. Kanbara, M., Yokoya, N.: Real-time estimation of light source environment for photorealistic augmented reality. In: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. vol. 2, pp. 911–914 Vol.2 (Aug 2004)
17. Lam, P.M., Ho, T.Y., Leung, C.S., Wong, T.T.: All-frequency lighting with multiscale spherical radial basis functions. IEEE Trans. Vis. Comput. Graph. 16(1), 43–56 (2010), http://dblp.uni-trier.de/db/journals/tvcg/tvcg16.html
18. Lourakis, M.I.A., Argyros, A.A.: Sba: A software package for generic sparse bundle adjustment. ACM Trans. Math. Softw. 36(1), 2:1–2:30 (Mar 2009), http://doi.acm.org/10.1145/1486525.1486527
19. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2. pp. 1150–. ICCV '99, IEEE Computer Society, Washington, DC, USA (1999), http://dl.acm.org/citation.cfm?id=850924.851523
20. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2. pp. 674–679. IJCAI'81, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1981), http://dl.acm.org/citation.cfm?id=1623264.1623280
21. Ng, R., Ramamoorthi, R., Hanrahan, P.: All-frequency shadows using non-linear wavelet lighting approximation. ACM Trans. Graph. 22(3), 376–381 (Jul 2003)
22. Ng, R., Ramamoorthi, R., Hanrahan, P.: Triple product wavelet integrals for all-frequency relighting. In: ACM SIGGRAPH 2004 Papers. pp. 477–487. SIGGRAPH '04, ACM, New York, NY, USA (2004)
23. Papagiannakis, G., Foni, R.: N.: Practical precomputed radiance transfer for mixed reality. In: In: Proceedings of Virtual Systems and Multimedia 2005. pp. 189–199. VSMM Society (2005)
24. Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., Fuchs, H.: The office of the future: A unified approach to image-based modeling and spatially immersive displays. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. pp. 179–188. SIGGRAPH '98, ACM, New York, NY, USA (1998), http://doi.acm.org/10.1145/280814.280861
25. Scaramuzza, D., Fraundorfer, F.: Visual odometry : Part i - the first 30 years and fundamentals. Robotics Automation Magazine, IEEE 18(4) (2011)
26. Shi, J., Tomasi, C.: Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on. pp. 593–600 (Jun 1994)
27. Sloan, P.P., Kautz, J., Snyder, J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In: Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques. pp. 527–536. SIGGRAPH '02, ACM, New York, NY, USA (2002)
28. Tsai, Y.T., Shih, Z.C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. In: ACM SIGGRAPH 2006 Papers. pp. 967–976. SIGGRAPH '06, ACM, New York, NY, USA (2006), http://doi.acm.org/10.1145/1179352.1141981
29. VR, O.: Oculus rift - virtual reality headset for 3d gaming. http://www.oculusvr.com/ (2012), accessed May 7, 2014
30. Wang, J., Ren, P., Gong, M., Snyder, J., Guo, B.: All-frequency rendering of dynamic, spatially-varying reflectance. In: ACM SIGGRAPH Asia 2009 Papers. pp. 133:1–133:10. SIGGRAPH Asia '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1661412.1618479