Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Article Type (Research/Review)

Adaptive Stray-Light Compensation in Dynamic Multi-Projection Mapping

Christian Siegl[1] ✉, Matteo Colaianni[1], Marc Stamminger[1] and Frank Bauer[1]

Abstract Projection based Mixed Reality is an effective tool to create immersive visualizations on real-world objects. Its wide range of applications includes art installations, education, stage shows and advertising. In this work, we enhance a multi-projector system for dynamic projection mapping by handling various physical stray-light effects: interreflection, projector black-level and environment light in real-time for dynamic scenes.
We show how all these effects can be efficiently simulated and accounted for at runtime, resulting in significantly improved projection mapping results. By adding a global optimization step, we can further increase the dynamic range of the projection.

Keywords mixed reality, projection mapping, multi-projector, real-time, interreflection, environment light.

1 Introduction

Projection Mapping setups are a popular way to alter the appearance of real-world objects, which is used in a wide range of applications. The system presented in this paper is based on the work by Siegl et al. [10]. Their multi-projection system tracks the target object in real-time using a depth camera. The blending between projectors is continuously adapted to the current object position and orientation. To compute the correct blending between projectors, a non-linear system – incorporating multiple projection quality terms – is solved on the GPU. With this system, a very high quality projection mapping is achieved at real-time rates on arbitrary white lambertian geometry.

However, their system ignores three key physical lighting effects, that can have significant impact on projection quality (see Figure 1):

- **Interreflection**: The indirect light produced by projecting on concave areas of white lambertian target geometry.
- **Black-Level**: The light a projector emits when set to present pure black. In particular when using LCD-projectors, this light is very noticeable.
- **Environment Light**: Low-frequency light that is cast by other, external sources.

All these effects result in too bright regions. In this paper we show how to simulate all these stray-light effects and compensate for them, by reducing the projected light accordingly (see Figure 1). This requires the real-time simulation of interreflections and environment lighting, for which we apply techniques from real-time rendering. When reducing the amount of projected light, we face the problem of loosing dynamic range for dark scenes and bright environments. By introducing an additional global optimization step, we can counteract this effect. Our adaptive algorithm noticeably improve the visual quality without a significant impact on performance.

2 Previous Work

The base work for understanding the interaction of light between diffuse surfaces was presented by Goral et al. [5]. Building on this, Sloan et al. [11] presented their work on real-time precomputed radiance transfer, which works very well for low frequency lighting. While we use their basic idea for compensating environment light, our setup is quite different. With light from multiple projectors, our lighting is dominated by

1 Computer Graphics Group, University of Erlangen-Nuremberg, Cauerstrasse 11, 91058 Erlangen, Germany. E-mail: Christian.Siegl@fau.de.

Manuscript received: 2017-02-22; accepted: 2017-02-22.

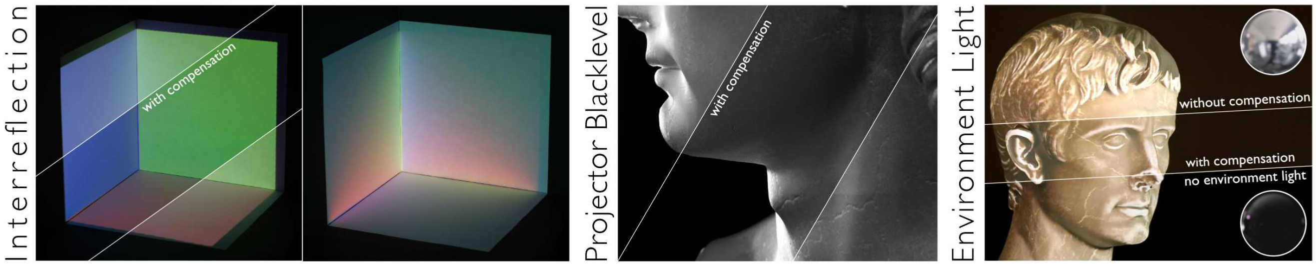


Fig. 1 Left: Correcting for interreflection in the corner of a box, where the sides are colored using projection mapping, along with an image of the indirect light. Middle: Correcting for the black-level of the projectors. Note how the brightness discontinuity on the neck disappears. Right: Correcting for daylight. For reference, the projection without environment light is captured at the bottom. Note the corresponding light probes. ***All images in this paper are pictures of real projections, captured with a DSLR.***

very bright spot lights invalidating the low frequency assumption.

The impact of scattered light from projection mapping was first described by Raskar et al. [8]. They argue, the scattered light contributes to the realism of the projection as it generates global illumination effects. For diffuse real-world- as well as diffuse virtual-materials this assumption is true. As a result they chose to ignore the scattered light in their projection. However, since we want to simulate different types of materials (i.e. glass, etc.), we need to eliminate the diffuse scattered light first.
The first compensation method for scattered light was shown by Bimber et al. [2]. They generate a precomputed irradiance map for a background scene. In contrast to our work, this map is restricted to static scenes – including the projected content.
Closer to our implementation is later work by Bimber et al. [3]. However, they show results only for planar and other trivially developable target surfaces. In addition, they use an expensive iteration scheme. We will show, that with a simplifying assumption this is not required. Similar to our light transport computations, Bermano et al. [1] solve the contribution from multiple projectors, while also accounting for subsurface scattering and projector defocus. However, their system does not compute the results in real-time. Yu et al. [9] present a method for correcting artifacts from interreflection based on perception. While they show promising results, their optimization scheme also does not run in real-time.

Another related field of research is the rendering of synthetic objects into real scenes (for an overview see [4]). Here, the main task is to estimate the environment light of the real scene from a plain RGB-image, to simulate the interaction with the rendered objects correctly. Since we change the appearance of the target object with the projection, we can no longer estimate the environment light directly from an image of the target geometry. Therefore, we use a light-probe as a proxy to gather the environment light.

Fig. 2 An exemplaric setup with two projectors, a depth camera for tracking and a diffuse white target object.

3 Base System

We use the system presented by Siegl et al. [10] as a basis for this work. Their system is able to solve the complex problem of blending multiple projectors on an arbitrary target geometry in real-time. The target object is tracked using a depth camera (marker-less tracking). Using the extrinsic and intrinsic information of the pre-calibrated system, the target object is then rendered from the viewpoint of the projectors. For blending, the system takes into account the target geometry and the expected projection quality. The resulting heuristic is based on the fact, that incident rays will give a sharper projection if they hit the target surface at a more perpendicular angle. Their system builds a non-linear optimization problem that incorporates the physical properties of light, the expected projection quality and a regularization term. The entire problem (represented as a transport matrix)

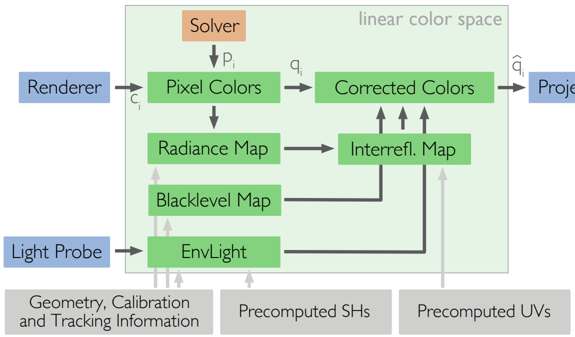


Fig. 3 An overview of the system.

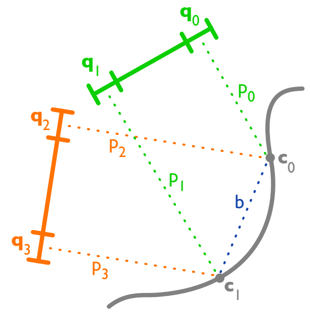


Fig. 4 The light scattering $\mathbf{b}$ between two surface points $\mathbf{c}_i$, illuminated by two projectors in a concave surface area.

is solved in real-time on the GPU, optimizing a base luminance p_i for every projector ray i (i addresses the pixel coordinates of all projectors sequentially). Projecting the base luminances by all projectors' pixels results in a uniform illumination. To generate a target image on the object, these luminances are modulated by the *target color* $\mathbf{c}_j$, resulting in the required pixel color $\mathbf{q}_i$ (for now, we assume a projector's luminance to be linear, which in practice is not the case):

$$\mathbf{q}_i = p_i \cdot \mathbf{c}_j \tag{1}$$

4 Real-Time Interreflection Correction

With multiple high powered projectors pointed at a white lambertian target object, surface points in non-convex regions receive light not only from the projectors, but also from their surrounding. Not accounting for this light results in too bright regions, which is visible in Figure 6. It would be possible to add this indirect illumination to the transport matrix of the previously described optimization problem by introducing additional matrix entries containing the indirect contributions of each projector ray. However, the highly increased number of non-zero entries makes solving the system much more expensive. In the following we describe a cheaper and equally powerful solution, leaving the transport matrix and thus the performance of the per-pixel luminance solver unchanged.

Since we want to examine the propagation of light between surface areas of the target object, we first need a parameterization of this object. We receive this parametrization by applying standard texture unwrapping algorithms commonly found in most 3D-modeling applications. Every texel i of the resulting texture corresponds to a surface point $\mathbf{x}_i$ with the associated normal $\mathbf{n}_i$.

To approximate the indirect irradiance $\mathbf{I}_i$ of a surface point, corresponding to texel i, we employ a standard technique from ray-tracing: We cast N rays from $\mathbf{x}_i$ in sample directions ω_i, which are cosine distributed on the hemisphere around $\mathbf{n}_i$:

$$\mathbf{I}_i = \frac{1}{N} \sum_{j=1\ldots n} \mathbf{C}(\mathbf{x}_i, \omega_j), \tag{2}$$

$\mathbf{C}(\mathbf{x}_i, \omega_j)$ is the target color at the surface point hit by the ray starting at $\mathbf{x}_i$ in direction ω_j. If the ray does not hit the object, $\mathbf{C}(\mathbf{x}_i, \omega_j)$ is black.

Since sampling the hemisphere during runtime would contradict our real-time requirements, we precompute the invariant locations of the surface intersections in texture space. In this preprocessing step, the hit points of the cosine weighted hemisphere samples are gathered for every texel. We then save the UV-coordinates of every hit point in a position lookup table. The indirect lighting computation is thus reduced to:

$$\mathbf{I}_i = \frac{1}{N} \sum_{j \in \mathcal{N}_i} \mathbf{c}_j, \tag{3}$$

$\mathcal{N}_i$ is the list of texels hit by the sample rays of texel i and $\mathbf{c}_j$ is the target color at texel j.

In convex regions the lists are empty, and also in concave regions usually only few sample rays hit the object. Therefore, $\mathcal{N}_i$ generally contains few elements, except for extreme cases. We further restrict the lists to the 64 rays with the most contribution. As a heuristic, we use Lambert's cosine law to determine the expected stray-light contribution. Moreover, the texture's resolution does not need to be very high, resulting in moderate precomputation times and memory consumption.

In projection mapping, our main objective is reproducing the desired target colors $\mathbf{c}_j$ at any surface point on the target object. We utilize the fact that, in contrast to general image generation, the exact target color/illumination at every surface point of the object is known. Under the assumption, that we are able to

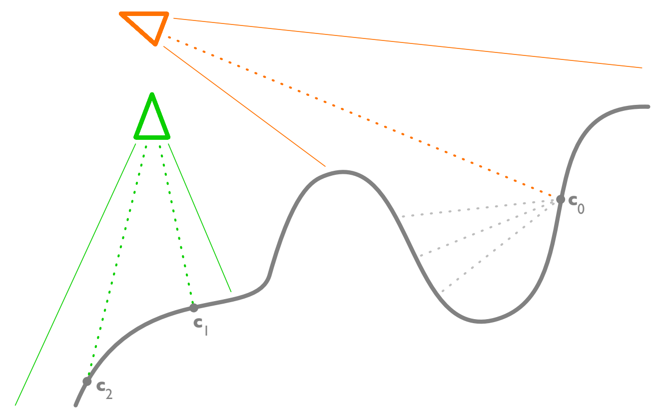


Fig. 5 Self shadowing and limited range of the projectors can lead to artifacts. $\mathbf{c}_0$ can only receive interreflecting light from the opposed surface if this surface is actually lit. Also the surface geometry influences the reachable surface illumination. While $\mathbf{c}_1$ is fully lit, $\mathbf{c}_2$ is attenuated due to Lambert's Law.

obtain this exact illumination on the target object, we may assume the illumination $\mathbf{I}_j$ to be *already present* at every surface location j. Using Equation 3, the indirect light $\mathbf{I}_j$ at every texel j can quickly be determined. This indirect light is then subtracted from each target color $\mathbf{c}_j$

$$\hat{\mathbf{q}}_i = p_i \cdot (\mathbf{c}_j - \mathbf{I}_j) \tag{4}$$

and sent to the projector.

An important additional implication of this approach is that we do not need to perform any iteration scheme.

4.1 Self Shadowing and Limited Range

With the algorithm described so far, we introduce artifacts due to

- self shadowing of the target geometry,
- lambertian and distance effects and
- limited range of the projectors.

All these effects cause certain areas of the illuminated object to not obtain the full target color. However, if we incorrectly assume that these areas contribute to interreflection with their full target color, we compensate for interreflected light that does not exist. This is demonstrated for surface point $\mathbf{c}_0$ in Figure 5. The opposing surface is not lit and therefore should not contribute to the interreflected light at $\mathbf{c}_0$.

To compensate for this, we compute a *Target-Color Map*. By gathering the contribution from all projectors at every surface point, we can compute the surface color that is achieved in the real world.

4.2 Implementation

In addition to sampling the hemisphere at every surface point and saving the resulting UV-coordinates in the preprocessing step, we also save the position and normal per surface point. This information is needed to compute the *Target-Color Map*.

The live system, generates the following information:

- *Target-Color Map*: Before we gather interreflected light using precomputed UV-coordinates (see Equation 3), we store the target colors in a *Target-Color Map*. In cases where the projection system cannot achieve the desired illumination due to physical limitations (self shadowing or lambertian law), the stored colors are attenuated accordingly.
- *Interreflection Map*: Using the *Target-Color Map* and the precomputed UV-coordinates, a second texture, containing the interreflected light (see Equation 3), is computed.

The final color sent to the projector is dimmed with the values from the *Interreflection Map* (see Equation 4). For a schematic of the implementation see Figure 3.

4.3 Linear Color Space

In general, addition and subtraction of colors are only valid in a linear color space. The colors in our processing pipeline are not linear but already have gamma applied. The same problem also affects the luminance values from the per-pixel solver, which determines blend values in linear space. Furthermore, the sum of color contributions and the subtraction from the final projected color are only valid in a linear color space.

This means, all incoming colors (from the renderer and the light probe) have to be linearized (inverse gamma). All computations are then performed within a linear color space (see Figure 3). Before sending the final color to the projector, we de-linearize the colors by applying a gamma correction. For a more detailed discussion we refer the reader to Siegl et al. [10].

5 Projector Black-Level

Another effect that impairs the quality of a projection mapping system, especially for dark scenes, is the projector's black-level. Even when projecting pitch black, the affected surface is brighter than it is when not projecting on it. We utilize LCD projectors for the benefit of reduced flickering when capturing the projections with a video camera. While the black-level

Fig. 6 Results for our interreflection correction. Notice the more even luminance distribution.

Fig. 7 Result for environment-light compensation.

of DLP-projectors in general offers a slightly better black-level, the problem is still very visible.

The previously introduced pipeline for interreflection correction is easily extensible for black-level correction. When computing the *Target-Color Map*, every texel of the surface texture is already reprojected into every projector and their contribution is gathered. At this stage a *Black-Level Map* is computed, gathering the cosine weighted incident black-level $\mathbf{B}_j$ from all projectors at every surface point. This incident black-level from all the projectors can – analogous to interreflected light – be interpreted as being already present on the target surface. As a result, $\mathbf{B}_j$ also has to be subtracted from the target surface color in the final rendering pass. Applying this correction to Equation 4 yields:

$$\hat{\mathbf{q}}_i = p_i \cdot (\mathbf{c}_j - \mathbf{I}_j - \mathbf{B}_j) \tag{5}$$

Since the exact black-level of the projectors is unknown, a calibration step is required. To estimate the value, a uniform grey illumination of 0.2 on the object is generated. Without black-level compensation, areas that are illuminated by only a single projector are noticeably darker than those where multiple projectors contribute (see Figure 1, middle). For calibration, the user then adjusts the black-level such that this difference disappears.

6 Environment Light

Another influence of unwanted light, affecting the projection quality, is environment light. Many projection mapping systems assume the environment to be perfectly dark. Of course, in a real setup this is generally not the case.

To counteract the influence of environment light in our dynamic real-time setup, we capture the surface of a mirrored hemisphere in real-time. The camera is intentionally defocused and only a low resolution image is acquired. Using this low frequency input, 9 spherical harmonics coefficients (per color channel) are computed from the light probe image (*SH vector*). For applying this information to the target color, an additional precomputation step is required: All hemisphere samples (see Section 4) missing the target object are projected into the space of the spherical harmonics base functions. With the resulting *transfer vector*, the incident environment light $\mathbf{E}_j$ is determined by the inner product of the *SH vector* and the *transfer vector* (per color channel, for further details see [11]).

We interpret this environment light $\mathbf{E}_j$ as an illumination that is already present on the target surface (similar to Sections 4 and 5). As a result, the projected color $\hat{\mathbf{q}}_i$ has to be reduced by $\mathbf{E}_j$, extending Equation 5 to:

$$\hat{\mathbf{q}}_i = p_i \cdot (\mathbf{c}_j - \mathbf{I}_j - \mathbf{B}_j - \mathbf{E}_j) \tag{6}$$

Fig. 8 Result of our global optimization step.

7 Global Optimization

With the presented method of offsetting every target surface color by an amount of light that can be interpreted as being already present on the surface, we very efficiently correct for artifacts from stray-light in projection mapping. However, one problem remains: If ambient, black-level or interreflected light exceeds the target illumination at a surface point, *negative light* would be required to achieve the target color. Obviously, this is impossible.

Solving a global non-linear optimization problem as introduced by Grundhoefer [6] would be a solution in this case. However, for our real-time system this is prohibitive from a performance point of view.

We propose a simpler and equally effective approach. By performing a scan operation over the final target colors $\hat{\mathbf{c}}_i$, we find the smallest target color component $\hat{c}^s$. This information is then used to offset the final projection as a whole, using a global correction factor C:

$$C = \begin{cases} C + (-C - \hat{c}^s) \times \alpha, & \text{if } \hat{c}^s \leq 0 \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Since C is computed in every frame, it can potentially change very fast based on the lighting and target projection colors. This raises the need for the dampening factor α, which was set to 0.1 in our examples. The value is dependent on the framerate and the expected variety in lighting and projected colors. Applying this factor ensures that the user will not notice the adaptions required to achieve the best projection quality possible.

The scalar global correction factor C is then applied to Equation 6:

$$\hat{\mathbf{q}}_i = p_i \cdot (\mathbf{c}_j - \mathbf{I}_j - \mathbf{B}_j - \mathbf{E}_j + \begin{bmatrix} C \\ C \\ C \end{bmatrix}) \tag{8}$$

By adding this scalar correction factor to all color channels we prevent any shifts in color. To prevent the system from failing under extreme lighting conditions, we restrict C to a maximum value of 0.25.

The effect of the global optimization step can be seen in Figure 8. On the right, the border between two projectors can no longer be compensated. While the black-level compensation works, in this especially dark area the final color would be required to be negative. With the global optimization on the left, the discontinuity disappears and the overall projection regains detail in the dark areas.

One additional problem is introduced with this approach: The amount of interreflected light changes due to the adapted colors. To correctly compensate this, an iteration scheme would be required. However, given that the lighting situation in general does not change rapidly, we supply the correction value to the next frame. Therefore, the projection will be correct within a few frames (also depending on α), which is not perceivable to the user. This is especially true, since changes in C only occur when the projection or the lighting conditions change noticeably. This draws more attention from the viewer than our small adjustments for improved projection quality.

8 Results and Discussion

Figure 1 shows results for all three presented compensation methods. The leftmost image shows the projection into the corner of a box, as well as an intensified image of the indirect light $\mathbf{I}$ we subtract when performing our correction (see Equation 3). Figure 6 and 9 show this compensation on a more complex surface. Note, how bright and discolored areas around hair, eyes and mouth disappear noticeably.

The middle image in Figure 1, demonstrates our black-level compensation. Note, the projection is too bright where two projectors illuminate the object. With our correction, the artifact disappears and the overall contrast of the projection improves.

The rightmost image in Figure 1 demonstrates the projection without (top) and with (middle) environment light compensation along with the captured light probes. For reference, the ground truth (a projection without any environment light) is shown

Fig. 9 Results for our interreflection correction. Notice the reduced color spill / increased contrast around the eye.

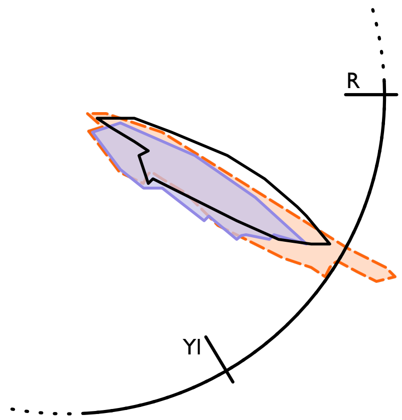


Fig. 10 The vectorscope for a projection without (black), uncompensated (orange, dashed) and compensated (blue) environment light.

at the bottom of the depicted bust. The difference between the corrected and uncorrected image in a room lit by daylight is immediately noticeable. Even with the addition of a large amount of environment light our corrected result is comparable to the ground truth. Only very dark regions are not completely compensated, since it is not possible to project *negative* light. This effect is best observed in our video, where we gradually open the shades and thereby increase the amount of environment light. The perceived dynamic range, contrast and color of the projection is constant due to our correction.

Enabling our global compensation mechanism in general is of benefit and improves the perceived color correctness of the projection. However this global step (increasing brightness) counteracts the dynamic environment compensation (decreasing brightness) under certain circumstances. In this paper we want to demonstrate the isolated effect of the environment compensation, thus we omit the global compensation in this section.

On the other hand Figure 7 shows a physical limitation of the presented system without a global step. Correction for a given surface effect is only possible, when the projectors physically have enough headroom left in terms of their dynamic range. For dark target colors, the incident light from interreflection, environment light and black-level may no longer be compensable, since it would be required to project *negative* light. This applies to a single color channel as well (regions with a pure or close to pure red, green or blue target color). The effect is noticeable on the door when comparing the compensated result (middle) with the ground-truth (bottom). The applied car paint is a very saturated, dark red. Thus, we can not compensate the green and blue contributions of the added environment light. In the region of the blinker light, where the target color is brighter and less saturated the compensation has the desired effect.

Figure 10 shows the vectorscope of the environment light compensation depicted in Figure 1. Black is the vectorscope of the bust for the original projection without environment light.When opening the shades, warm, yellow environment light is added. The orange segment (dashed outline) in the scope represents our projection without any compensation. This results in a strong peak on the red side of the segment as well as a general increase in brightness (radial scale of the segment). The blue segment depicts results with our compensation turned on. The general intensity and color distribution closely resembles the ground truth (black segment). It is also noticeable that the orange and blue segments shape towards the center of the scope is broader due to added environment light on the background of the image.

8.1 Setup

Our hardware setup consists of a an off-the-shelf workstation: `Intel Core i7 4771 (3.5GHz)` and `NVidia GeForce GTX 980` graphics card. As projectors two `NEC NP-P451WG`, with a resolution of 1280 by 800 pixels are calibrated to an `ASUS Xtion PRO`

	Augustus (25k faces)	Truck (300k faces)
Tracking	1.1 ms	1.3 ms
Rendering	9.4 ms	12.9 ms
Target-C. / Black-L. Map	1.1 ms	0.9 ms
Interreflection Map	0.9 ms	0.7 ms
Per-Pixel Solver	8.5 ms	9.0 ms
Overall GPU Time	21.0 ms	24.8 ms
Spherical Harmonics (CPU)	17 ms	17 ms
Frame Rate	$\sim$ 40 fps	$\sim$ 37 fps

Tab. 1 Per-frame performance numbers for all parts of our algorithm. Everything runs on the GPU, except the spherical harmonics calculation, which run in a concurrent CPU thread.

`Live` depth sensor for object tracking. An exemplary setup is depicted in Figure 2.

8.2 Performance

Given a good parametrization of the object, a texture resolution of 1024×1024 for the precomputed data, *target-color*, *interreflection* and *black-level map* proved to be sufficient in our experiments. For the hemisphere, 64 samples showed good results. These numbers depend on the target object and the quality of the surface parametrization. Texture resolution mainly depends on the size of the target object and the unwrapping quality. The object complexity is the defining factor for the number of samples. Since the treated effects are rather low in frequency, these values can be chosen rather conservatively.

For detailed performance numbers on the generation of the maps, see Table 1. Applying the correction to the final projected color has no measurable performance impact. Computing the spherical harmonics coefficients is performed in a concurrent CPU thread in real-time and does not impact performance or latency. Since the preprocessing step runs offline, its performance is non-critical.

Given the tight time constraint, certain data is precomputed (uv-coordinates for interreflections, transfer vectors for spherical harmonics). As a result we can only project on non-deformable geometry. However, due to the runtime calculations, the target object can be moved and the projected content can change dynamically. This is especially important for animations on the target objects or a live painting system as presented by Lange et al. [7].

9 Conclusion

In this paper we have presented a fast and reliable system for correcting projection mapping artifacts in dynamic scenes from interreflection, projector black-level and environment light. To meet the tight time constraints of a low latency projection mapping system, some data is precomputed. Paired with the important assumption of knowing the exact color of every surface point, correcting artifacts from unwanted lighting is performed efficiently during runtime. With these extensions, the perceived quality of any projection mapping system is improved significantly.

References

[1] A. Bermano, P. Brüschweiler, A. Grundhöfer, D. Iwai, B. Bickel, and M. Gross. Augmenting physical avatars using projector-based illumination. *ACM Trans. Graph.*, 32(6):189:1–189:10, Nov. 2013.

[2] O. Bimber, A. Grundhöfer, G. Wetzstein, and S. Knödel. Consistent illumination within optical see-through augmented environments. In *Proceedings of the 2Nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '03, Washington, 2003.

[3] O. Bimber, A. Grundhofer, T. Zeidler, D. Danch, and P. Kapakos. Compensating indirect scattering for immersive and semi-immersive projection displays. In *IEEE Virtual Reality Conference (VR 2006)*, pages 151–158, March 2006.

[4] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, New York, NY, USA, 2008. ACM.

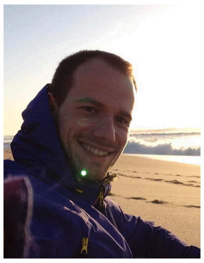
[5] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile. Modeling the interaction of light between diffuse surfaces. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 213–222, New York, NY, USA, 1984. ACM.

[6] A. Grundhöfer. Practical non-linear photometric projector compensation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on.* IEEE, 2013.

[7] V. Lange, C. Siegl, M. Colaianni, P. Kurth, M. Stamminger, and F. Bauer. Interactive painting and lighting in dynamic multi-projection mapping. In Springer, editor, *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*, pages 113–125. Springer, 2016.

[8] R. Raskar, G. Welch, K.-L. Low, and D. Bandyopadhyay. Shader lamps: Animating real objects with image-based illumination. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, London, 2001. Springer-Verlag.

[9] Y. Sheng, B. Cutler, C. Chen, and J. Nasman. Perceptual global illumination cancellation in complex projection environments. *Computer Graphics Forum*, 30(4):1261–1268, 2011.
[10] C. Siegl, M. Colaianni, L. Thies, J. Thies, M. Zollhöfer, S. Izadi, M. Stamminger, and F. Bauer. Real-time pixel luminance optimization for dynamic multi-projection mapping. *ACM Transactions on Graphics (TOG)*, 34(6), 2015.
[11] P.-P. Sloan, J. Kautz, and J. Snyder. 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*, SIGGRAPH '02, pages 527–536, New York, NY, USA, 2002. ACM.



Christian Siegl is a PhD candidate at the Computer Graphics Group of the University of Erlangen-Nuremberg. His research is focused on mixed-reality using projection mapping, medical image processing and the virtual creation of apparel.

Matteo Colaianni is a PhD candidate at the Computer Graphics Group of the University of Erlangen-Nuremberg. His focus of research is geometry processing in the field of apparel development as well as statistical shape analysis.

Frank Bauer is a research fellow at the Computer Graphics Group of the University of Erlangen-Nuremberg. His research is focused on 3D scene reconstruction, augmented, mixed and virtual-reality applications as well as accessible human-machine interactions.

Marc Stamminger is a professor for visual computing at the Computer Graphics Group of the University of Erlangen-Nuremberg since 2002. His research is focused on real-time visual computing, in particular rendering, visualization, interactive computer vision systems, augmented and mixed reality.