Computational Visual Media
DOI 10.1007/s41095-017-0090-8 Vol. 3, No. 3, September 2017, 263–271

Research Article

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 environmental 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; environmental light

1 Introduction

Projection mapping setups are a popular way to alter the appearances of real-world objects, and are used in a wide range of applications. The system presented in this paper is based on the work by Siegl et al. [1]. Their multi-projection system tracking the target object in 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, Siegl et al.'s system ignores three key physical lighting effects that can have significant impact on projection quality (see Fig. 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.
- **Environmental light**: Low-frequency light that is cast by other external sources.

All these effects result in over-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 Fig. 1). This requires the real-time simulation of interreflections and environmental lighting, for which we apply techniques from real-time rendering. When reducing the amount of projected light, we face the problem of losing dynamic range for dark scenes and bright environments. By introducing an additional global optimization step, we can counteract this effect. Our adaptive algorithm noticeably improves the visual quality without a significant impact on performance.

2 Previous work

The basis for understanding the interaction of light with diffuse surfaces was presented by Goral et al. [2]. Building on this, Sloan et al. [3] presented work on real-time precomputed radiance transfer, which works very well for low-frequency lighting. While we use their basic idea for compensating for environmental light, our setup is quite different. With light from multiple projectors, our lighting is

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

Manuscript received: 2017-03-03; accepted: 2017-04-29

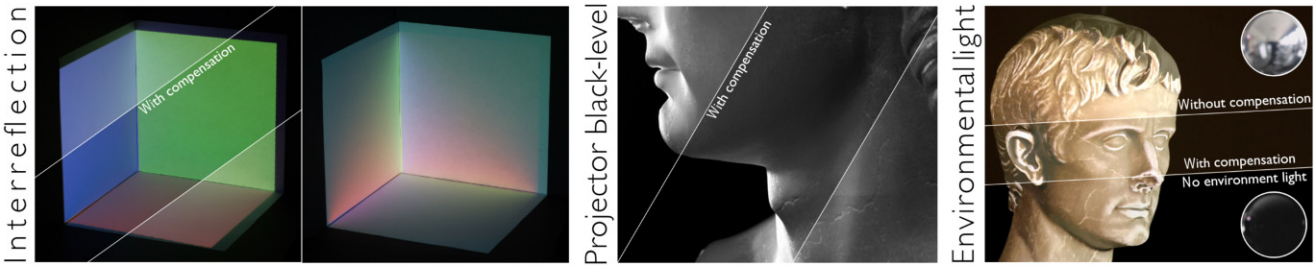


Fig. 1 Left: correcting for interreflection in the corner of a box, where the sides are colored using projection mapping, along with an image of the indirect light. Middle: correcting for the black-level of the projectors. Note how the brightness discontinuity on the neck disappears. Right: correcting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

dominated by very bright spot lights invalidating the low-frequency assumption.

The impact of scattered light from projection mapping was first discussed by Raskar et al. [4]. They argued that scattered light contributes to the realism of the projection as it generates global illumination effects. For diffuse real-world lighting as well as diffuse lighting of 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 (glass, etc.), we need to eliminate the diffuse scattered light first.

The first compensation method for scattered light was given by Bimber et al. [5]. They generated 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 approach is later work by Bimber et al. [6]. However, they showed results only for planar and other trivial developable target surfaces. In addition, they used an expensive iteration scheme. We show that, with a simplifying assumption, this is not required. As in our light transport computations, Bermano et al. [7] solved 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. Sheng et al. [8] presented a method for correcting artifacts from interreflection based on perception. While they showed promising results, their optimization scheme also does not run in real time.

Another related field of research concerns the rendering of synthetic objects into real scenes; for an overview see Ref. [9]. Here, the main task is to estimate the environmental lighting 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 environmental light directly from an image of the target geometry. Therefore, we use a light-probe as a proxy to gather the environmental light.

3 Base system

We use the system presented by Siegl et al. [1] 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 (without markers) using a depth camera. Using the extrinsic and intrinsic information of the pre-calibrated system, the target object is then rendered from the viewpoint of the projectors. During 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 is

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

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

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

4 Real-time interreflection correction

With multiple high powered projectors pointed at a white Lambertian target, surface points in non-convex regions receive light not only from the projectors, but also from their surroundings. Not accounting for this light results in regions which are too bright, as can be seen for example in Fig. 8. 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 greatly increased number of non-zero entries makes solving the system much more expensive. In the following we describe a cheaper but equally powerful solution, leaving the transport matrix unchanged, and hence also the performance of the per-pixel luminance solver.

Since we wish to examine the propagation of light between surface areas of the target object, we first need a parameterization of this object.

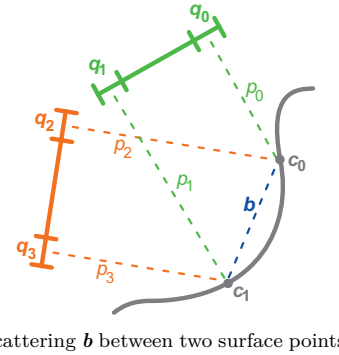


Fig. 3 Light scattering $\boldsymbol{b}$ between two surface points $\boldsymbol{c}_i$, illuminated by two projectors in a concave surface area.

We determine this parameterization by applying standard texture unwrapping algorithms commonly used in 3D-modeling applications. Every texel i of the resulting texture corresponds to a surface point $\boldsymbol{c}_i$ with associated normal $\boldsymbol{n}_i$.

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

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

where $\boldsymbol{C}(\boldsymbol{x}_i, \boldsymbol{\omega}_j)$ is the target color at the surface point hit by the ray starting at $\boldsymbol{c}_i$ in direction $\boldsymbol{\omega}_j$. If the ray does not hit the object, $\boldsymbol{C}(\boldsymbol{x}_i, \boldsymbol{\omega}_j)$ 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

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

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

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

In projection mapping, our main objective is to reproduce the desired target colors $\boldsymbol{c}_j$ at each surface point on the target object. We utilize the fact that, in contrast to general image generation, the exact target color and illumination at every surface point of the object are known. Given the assumption that we can obtain the exact illumination on the target object, we may assume the illumination $\boldsymbol{I}_j$ to be *already present* at every surface location j. Using Eq. (3), the indirect light $\boldsymbol{I}_j$ at every texel j can

quickly be determined. This indirect light is then subtracted from each target color $\boldsymbol{c}_j$:

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

and sent to the projector.

An important additional implication of this approach is that iteration is not needed.

4.1 Self shadowing and limited range

The algorithm described so far produces artefacts 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 fail to 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 $\boldsymbol{c}_0$ in Fig. 4: the opposing surface is not lit and therefore should not contribute to the interreflected light at $\boldsymbol{c}_0$.

To compensate for this, we compute a *target-color map*: by gathering the contributions from all projectors at each surface point, we can compute the surface color that results 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*.

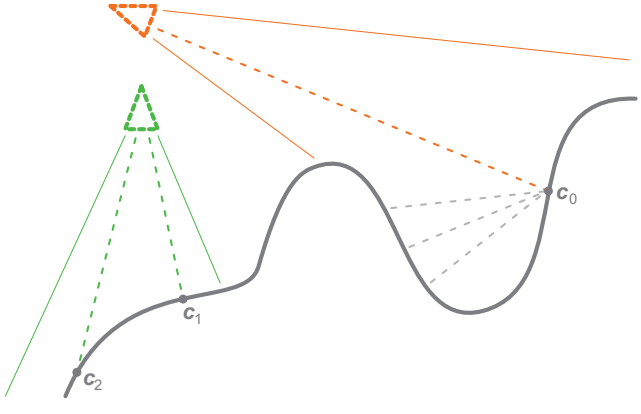


Fig. 4 Self shadowing and limited range of the projectors can lead to artifacts. $\boldsymbol{c}_0$ can only receive interreflecting light from the opposed surface if this surface is actually lit. Furthermore, the surface geometry influences the reachable surface illumination. While $\boldsymbol{c}_1$ is fully lit, $\boldsymbol{c}_2$ is attenuated due to Lambert's Law.

The live system generates the following information:

- *Target-color map*: Before we gather interreflected light using precomputed UV-coordinates (see Eq. (3)), we store the target colors in a *target-color map*. In the 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 Eq. (3)), is computed.

The final color sent to the projector is dimmed with the values from the *interreflection map* (see Eq. (4)). For a schematic of the implementation, see Fig. 5.

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 subtraction from the final projected color are only valid in a linear color space.

This means that all incoming colors (from the renderer and the light probe) have to be linearized (using inverse gamma transformation). All computations are then performed in a linear color space (see Fig. 5). Before sending the final color to the projector, we de-linearize the colors by applying gamma correction. For a more detailed discussion we refer the reader to Siegl et al. [1].

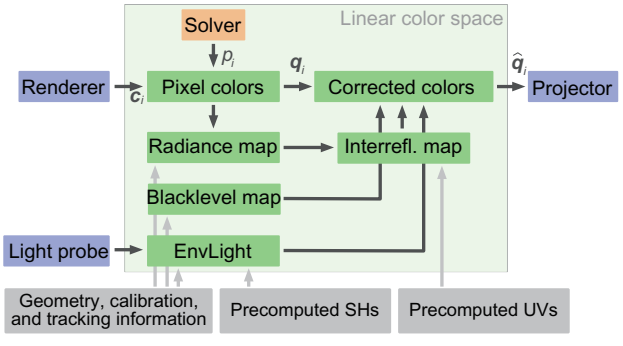


Fig. 5 An overview of the system.

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 pitches black, the affected surface is brighter than 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 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 to include 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 $\boldsymbol{B}_j$ from all projectors at every surface point. This incident black-level from all the projectors can, in an analogous way to interreflected light, be interpreted as being already present on the target surface. As a result, $\boldsymbol{B}_j$ also has to be subtracted from the target surface color in the final rendering pass. Applying this correction to Eq. (4) yields:

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

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 to which multiple projectors contribute (see Fig. 1, middle). For calibration, the user adjusts the black-level such that this difference disappears.

6 Environmental light

Another influence causing unwanted light, affecting the projection quality, is environmental 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 environmental 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 harmonic coefficients (per color channel) are computed from the light probe image (the *SH vector*). To apply 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 harmonic basis functions. The resulting *transfer vector* allows the incident environmental light $\boldsymbol{E}_j$ to be determined from the inner product of the *SH vector* and the *transfer vector* (per color channel; for further details see Ref. [3]).

Fig. 6 Environmental light compensation.

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

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

7 Global optimization

Using 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 discussed in Grundhöfer [10] is a solution to this problem. However, for our real-time system the performance of this approach is prohibitive. We propose a simpler and equally effective approach. By performing a scan over the final target colors $\hat{\boldsymbol{c}}_i$, we find the smallest target color component $\hat{c}^{\mathrm{s}}$. This information is then used to offset the final projection as a whole, using a global correction factor C:

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

Since C is computed for every frame, it can potentially change very quickly based on the lighting and target projection colors. This raises the need for a damping factor α, set to 0.1 in our examples. Its value is dependent on framerate and the expected variation 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 applied to Eq. (6):

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

By adding this scalar correction factor to all color channels, we prevent any shift 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 Fig. 7. On the right, the border between two projectors can no longer be compensated. While black-level compensation works, in this especially dark area the final color would need to be negative. Using 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 for this, an iterative 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 become correct within a few frames (also depending on α), which is not perceivable to the user. This is further helped as changes in C only occur when the projection or lighting conditions change noticeably. These draw more attention from the viewer than our small adjustments for improved projection quality.

Fig. 7 Results of our global optimization step.

8 Results and discussion

Figure 1 shows results for all three presented compensation methods. The leftmost image shows projection into the corner of a box, as well as an intensified image of the indirect light $\boldsymbol{I}$ we subtract when performing our correction (see Eq. (3)). Figures 8 and 9 show this compensation on a more complex surface. Note how bright and discolored areas around the hair, eyes, and mouth noticeably

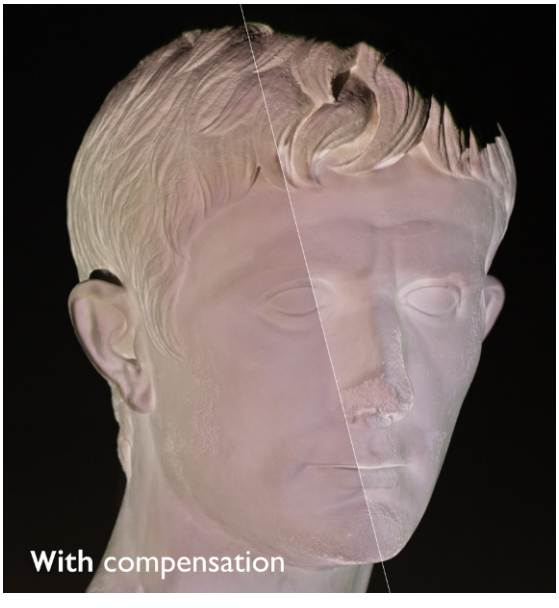


Fig. 8 Results of interreflection correction. Note the more even luminance distribution.

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

disappear.

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

The rightmost image in Fig. 1 demonstrates the projection without (top) and with (middle) environmental light compensation, along with the captured light probes. For reference, the ground truth (a projection without any environmental light) is shown 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 environmental 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 (https://vimeo.com/188121147), where we gradually open the shades and thereby increase the amount of environmental light. The perceived dynamic range, contrast, and color of the projection are 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 environmental compensation (decreasing brightness) under certain circumstances. Here, we want to demonstrate the isolated effect of the environmental compensation, so we omit the global compensation in this section.

On the other hand, Fig. 6 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, environmental light, and black-level may no longer be compensatable, since it would require the projection of *negative* light. This also applies to a single color channel (regions with a pure color, or one close to a 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 for green and blue contributions to the added environmental 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 a vectorscope of the environmental light compensation depicted in Fig. 1. Black is the vectorscope of the bust for the original projection without environmental light. When opening the shades, warm, yellow environmental light is added. The orange segment (dashed outline) in the scope represents our

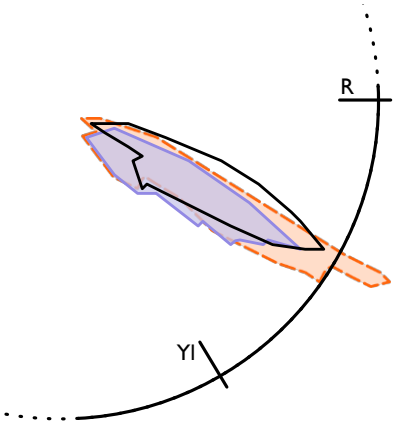


Fig. 10 The vectorscope for a projection without (black), uncompensated (orange, dashed), and compensated (blue) environmental light.

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 resemble the ground truth (black segment). It is also noticeable that the shapes of the orange and blue segments towards the center of the scope is broader due to added environmental light on the background of the image.

8.1 Setup

Our hardware setup consists of an off-the-shelf workstation, and Intel Core i7 4771 (3.5 GHz) with an NVidia GeForce GTX 980 graphics card. As projectors, two NEC NP-P451WG, with a resolution of 1280 × 800 pixels were calibrated to an ASUS Xtion PRO Live depth sensor for object tracking. A possible setup is depicted in Fig. 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 suffice 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. The 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 when generating maps, see Table 1. Applying the correction to the final projected color has no measurable performance impact. Computing the spherical harmonic 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.

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

	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	∼ 40 fps	∼ 37 fps

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 such as the one in Lange et al. [11].

9 Conclusions

In this paper we have presented a fast and reliable system for correcting projection mapping artefacts in dynamic scenes arising from interreflection, projector black-level, and environmental 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 can be improved significantly.

References

[1] Siegl, C.; Colaianni, M.; Thies, L.; Thies, J.; Zollhöfer, M.; Izadi, S.; Stamminger, M.; Bauer, F. Real-time pixel luminance optimization for dynamic multi-projection mapping. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 237, 2015.

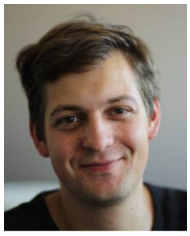
[2] Goral, C. M.; Torrance, K. E.; Greenberg, D. P.; Battaile, B. Modeling the interaction of light between diffuse surfaces. In: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques, 213–222, 1984.

[3] 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, 527–536, 2002.

[4] Raskar, R.; Welch, G.; Low, K. L.; Bandyopadhyay, D. Shader lamps: Animating real objects with image-based illumination. In: *Rendering Techniques*

2001. Gortler, S. J.; Myszkowski, K. Eds. Springer Vienna, 89–102, 2001.
[5] Bimber, O.; Grundhöfer, A.; Wetzstein, G.; Knodel, S. Consistent illumination within optical see-through augmented environments. In: Proceedings of the 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality, 198–207, 2003.
[6] Bimber, O.; Grundhöfer, A.; Zeidler, T.; Danch, D.; Kapakos, P. Compensating indirect scattering for immersive and semi-immersive projection displays. In: Proceedings of the IEEE Virtual Reality Conference, 151–158, 2006.
[7] Bermano, A.; Brüschweiler, P.; Grundhöfer, A.; Iwai, D.; Bickel, B.; Gross, M. Augmenting physical avatars using projector-based illumination. *ACM Transactions on Graphics* Vol. 32, No. 6, Article No. 189, 2013.
[8] Sheng, Y.; Cutler, B.; Chen, C.; Nasman, J. Perceptual global illumination cancellation in complex projection environments. *Computer Graphics Forum* Vol. 30, No. 4, 1261–1268, 2011.
[9] 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 ACM SIGGRAPH 2008 Classes, Article No. 32, 2008.
[10] Grundhöfer, A. Practical non-linear photometric projector compensation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 924–929, 2013.
[11] Lange, V.; Siegl, C.; Colaianni, M.; Kurth, P., Stamminger, M.; Bauer, F. Interactive painting and lighting in dynamic multi-projection mapping. In: *Augmented Reality, Virtual Reality, and Computer Graphics*. De Paolis, L.; Mongelli, A. Eds. Springer Cham, 113–125, 2016.

Christian Siegl is a Ph.D. 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 Ph.D. 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.

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, and augmented and mixed reality.

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.