Frameless

Volume 4 | Issue 1 Article 33

October 2021

Real-Time Illumination Capture and Rendering on Mobile Devices

Snehal Padhye
Rochester Institute of Technology

James A. Ferwerda
Rochester Institute of Technology

FRAMELESS

Real-Time Illumination Capture and Rendering on Mobile Devices

Snehal Padhye*
Rochester Institute of Technology

James A. Ferwerda
Rochester Institute of Technology

Abstract: *We present our efforts to develop methods for rendering 3D objects on mobile devices using real-world dynamic illumination from the user's environment. To achieve this, we use the front and back cameras on the mobile device to estimate the light distribution in the environment in real time. We then create a dynamic illumination map and render the object at interactive rates in a browser on the device using a web-based graphics API. This project achieves one of the goals of our related work on realistic visualization of virtual objects: to make virtual objects appear to be situated within the scene they are observed in.*

Keywords: *Illumination capture, 3D rendering, low-power mobile devices*

*Corresponding Author, Snehal Padhye
Published online April 15th, 2022

I. INTRODUCTION

Virtual objects represented by 3D models can be easily rendered and interacted with using modern computer graphics tools. The appearance of the virtual objects is due to effects of interaction of light with the surfaces of the objects. In order to realistically represent the appearance of an object, it is often rendered using a virtual illumination map representing the real-world environment. To make the virtual object appear like it is a part of the real environment, we need to use the real- world illumination map to render the virtual scene.

In previous work (Ferwerda 2014)(Ferwerda and Darling 2013) we have defined three criteria that must be met for a virtual object to appear to be part of the real world. First, the rendered images of the object must be realistic. Second, the system rendering the object must be responsive and allow natural interaction. And finally, the object must appear to be situated within the real environment. Accurate real-time modeling and rendering of the user ‘s illumination environment is key to this final goal.

The appearance of an object in a virtual scene is obtained by simulating how light is reflected or transmitted by the object. The appearance is made more realistic by calculating the global illumination which represents both the direct and indirect components of light transport. To make the rendered object more situated, we need to capture the user ‘s illumination environment in order to use it in our virtual scene.

While there has been lots of research on realistic rendering of objects in captured illumination environments, they usually require sophisticated setups and precomputations to estimate a given environment (Boddington and Thatte 2015)(Currius, et al. 2020). Our goal in this paper is to take advantage of the capabilities of modern mobile devices to capture a user ‘s illumination environment in real-time and to use web- based 3D graphics to realistically render 3D objects on the device so that they appear to be part of the user ‘s environment.

II. RELATED WORK

Realistic real-time rendering of 3D objects requires identifying illumination sources in the real-world scene, evaluating their direct and indirect contribution to the global illumination in the scene and keeping track of the sources and their effects on the objects over time. Researchers have developed various techniques to achieve these goals and one of them is to estimate the global illumination using basis functions. Wong et al (Wong, et al. 1997) used spherical harmonics to estimate the illumination, Ng et al (Ng, Ramamoorthi and Hanrahan 2003) used wavelets to approximate the illumination and more recently, Currius et al (Currius, et al. 2020) used a convolutional neural network to calculate parameters of spherical gaussian to approximate the illumination.

Image-based lighting has also become a popular method for global illumination estimation. The work can be traced back to Blinn (Blinn and Newell 1976) who used synthesized images to render appearance of an object. This led to the idea of creating a real-world scene by using images of the scene as an illumination map. Debevec (Debevec 2008) used HDR image-based model to provide realistic appearance to the object. To capture illumination environments in real time, developers have used cameras

with fish-eye lenses and large fields of view (Yoo and Lee 2008). Walton et al (Walton, et al. 2017) used an RGBD camera along with a fish-eye lens camera to construct user ‘s environment at real time. Panoramic approaches for reconstructing a 360-degree scene includes algorithms that use invariant features to stitch multiple image (Brown and Lowe 2007)(Iorns and Rhee 2016) to form an illumination map of the environment. More recently, deep learning methods have also been used to estimate illumination in a scene using a single image (Gardner, et al. 2017) (Liu, et al. 2020).

Capturing the real-time lighting environment on a mobile device is challenging. While all these methods represent important advances, not all of them can be used directly for realistic real-time rendering on mobile devices. Some methods require specialized hardware for capture. Others require precomputation of one or more components of the light field. To capture incident illumination, it is difficult to rely on the device user to have mirror like spherical objects in the scene. Mobile devices usually do not come with SLAM (Simultaneous Localization and Mapping) and light field sensors, but most certainly are always equipped with one or more cameras. Thus, image-based lighting looks like a promising approach to extend it on mobile devices. But the cameras on the devices have a narrow field of view and are low in resolution which is discouraging in constructing a panoramic HDR image. Additionally, image-based techniques do not capture dynamically varying illumination. The aim of our work is to develop methods to construct real time environment by using image-based lighting within the constraints of the mobile device. We also aim to incorporate real-time capture of the dynamic illumination environments and the realistic rendering of virtual objects using these illumination environments in real-time on standard mobile devices.

III. CURRENT SYSTEM

Our system has two major components: illumination capture and tracking and scene rendering. To capture and track the illumination in a scene, we use the mobile device ‘s inbuilt cameras. To support real-time rendering, we use Three.js (Three.js n.d.), which is a JavaScript library built on WebGL (WebGL n.d.). WebGL has brought the power of complex 3D graphics to web browsers and every modern browser supports it. Since, web browsers are ubiquitous in mobile devices, building a browser-based application enables widespread use of our system without the need for a user to download any code. The system can be accessed simply by loading a URL into a browser.

A typical Three.js pipeline for 3D graphics is shown in Figure 1. An object is characterized by its geometry and material properties. The geometry can be planar, spherical, or parametric. The material can vary in reflectance, transmittance, and emittance, and any of these properties as well as mesoscale texture can be modified spatially using image-based maps. These properties together form a Mesh that is added to a Three.js Scene. The Scene also requires specification of the illumination, and the parameters of a virtual camera that views the scene. Using this information, the Scene is rendered using the WebGL renderer and the resulting images are displayed in the browser.

In our system, we use image-based lighting techniques to specify the scene illumination. The back camera of the mobile device is used

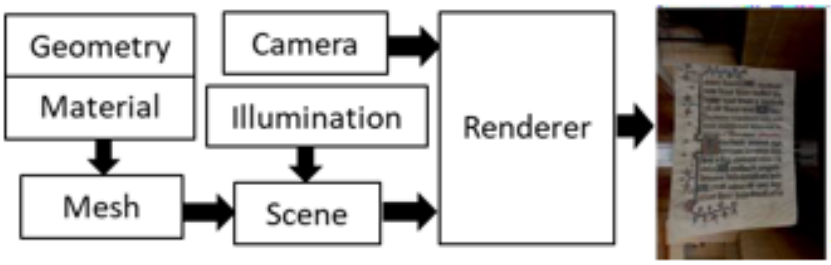

Fig. 1. Three.js rendering pipeline. An object defined by its Geometry and Material is used to form a Mesh. The Mesh is added to the Scene, and along with a representation of the illumination and a virtual camera is given to a renderer to produce realistic images of the scene at interactive rates.

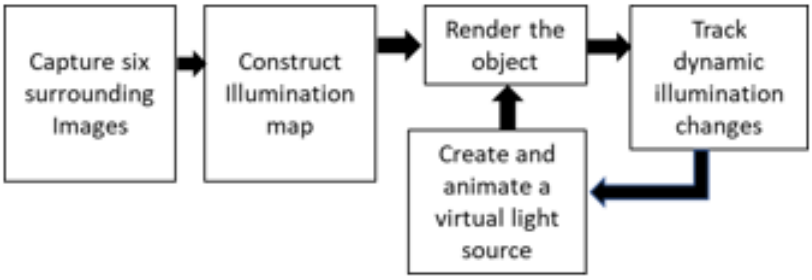

Fig. 2. Workflow to render a virtual object in an illumination environment. Using the device 's back camera, six images are captured to construct an illumination map of the user 's environment. The object is rendered with this illumination map, and real-time dynamic illumination sources are tracked by the front camera of the device. A virtual light source is created and animated according to changes in the real-world illumination. Although it is not apparent in the illustration, the fact that the illumination in the virtual environment matches the illumination in the user 's actual environment greatly increases the sense that the virtual object is situated in the user's real space.

to capture the illumination field in the user's environment, and the front camera is used to track dynamic changes in illumination field.

A. Capturing and rendering scene illumnation

There are various ways to construct an illumination map of an environment. We create our illumination representation using a cube map. The map can be defined by sampling the user's surround with six images that form six faces of a cube. Once the cube has been constructed, it can be used to illuminate objects in the scene. The overall flow of this processing in our system is shown in Figure 2. After launching the application, the user is asked to capture six images at roughly orthogonal angles that represent their real-world environment. These images are then used to create the illumination cube map which is then used to illuminate the object in the scene. Figure 3 shows an example of the images captured and an object rendered using this method. Figure 4 shows the effects of rotating the object within this captured illumination environment.

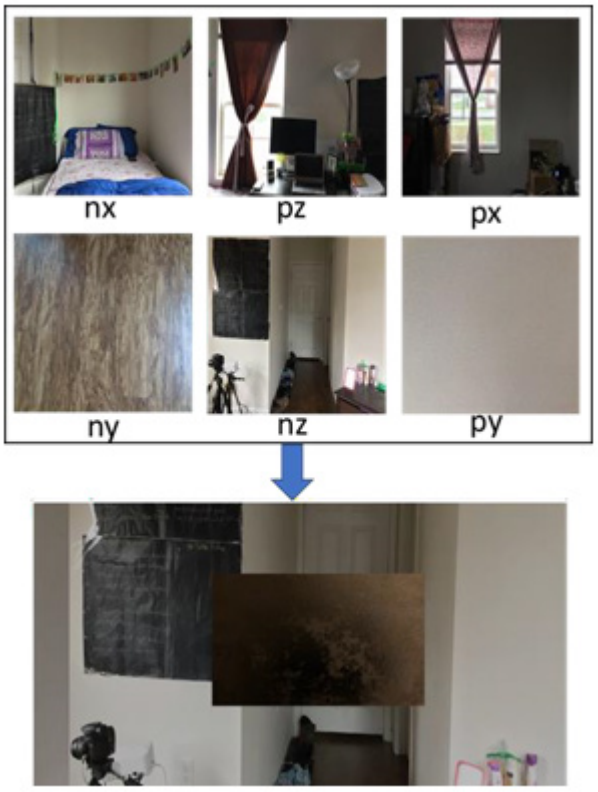

Fig. 3. An example of the six images captured along three orthogonal axes and a painting rendered using the constructed illumination map.

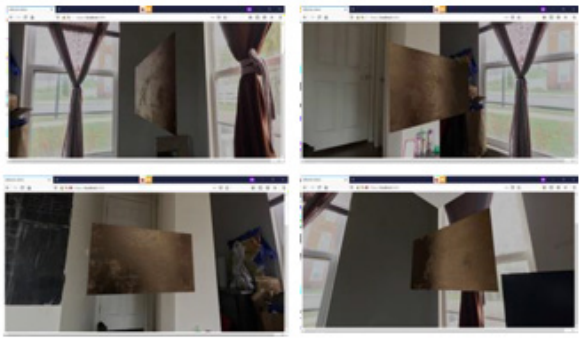

Fig. 4. Effects on object appearance of the captured illumination map at different viewing angles. The two windows in the apartment are the major sources of illumination in the scene and their effect is seen on the rendered object.

B. Tracking scene illumination changes

After constructing the cube map to represent the user‘s illumination environment, we capture frames periodically from the front facing camera of the mobile device to detect dynamic changes in illumination. To support this functionality, we use the OpenCV (OpenCV n.d.) JavaScript library to perform real-time image processing for dynamic tracking.

In our implementation, we use the front (user) facing camera to look for significant changes in scene illumination. Using OpenCV we estimate the direction of the change and place a virtual light source in the same relative direction in the scene model. The virtual light is added, modulated, displaced, and removed in concert with changes in the real-world environment. Figure 5 shows examples of this system capability. An object is rendered using an illumination map created from the user‘s environment, and the dynamic effects on the object ‘s appearance, of adding a moving flashlight to this environment are shown in the Figure.

IV. WORK IN PROGRESS

While the system described above gives good results, there is room of improvement on two aspects. First, even though the illumination map is created at run- time, the user needs to explicitly capture the six images to create the map. Second, due to the narrow fields of view of the device ‘s cameras, light sources in the scene lying outside the fields of view may not get tracked and the sampling of the environment into six images to create the cube map may not be sufficient. To overcome these shortcomings, we are currently working on a new system that represents the illumination

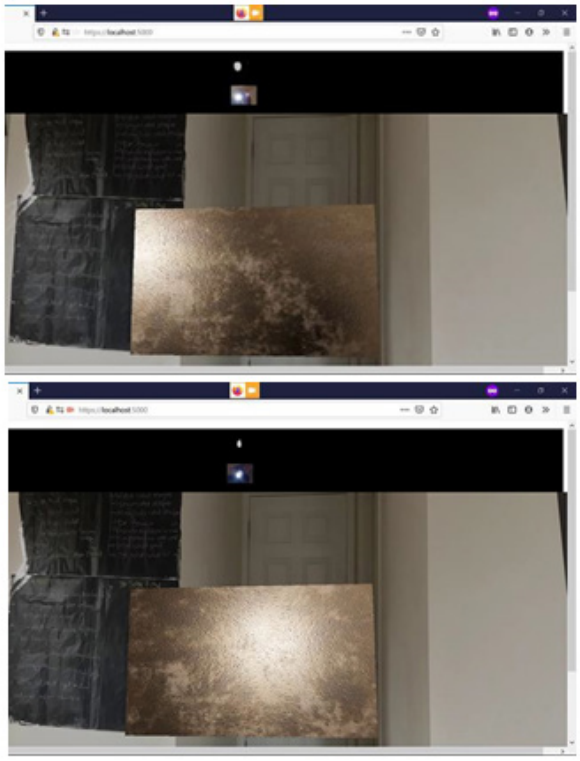

Fig. 5. Examples of effects of external light sources. Each browser window shows a copper surface rendered by our system. At the top of each window are two small frames showing the user pointing a flashlight at the virtual surface and the light source segmented from this frame. This segmented source is then used to create a virtual light source that dynamically illuminates the virtual surface.

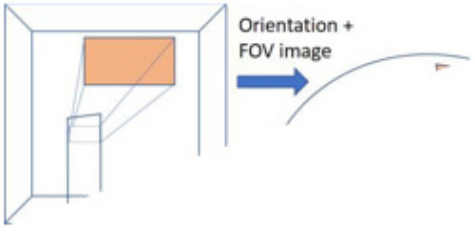

Fig. 6. The real time scene covered by the field of view of the camera is mapped to the rendered scene with hemispherical environment such that the direction and orientation of the camera is consistent with the user.

environment as an inner surface of a large hemisphere and divides it into tiles, based on fields of view of the cameras. As shown in Figure 6, for each view of the camera, the image is mapped to the respective tile of the hemisphere. To align the image to the tiles, data from the device ‘s orientation sensor is used. The mapping will be done at real time, and the map will be updated continuously to account for multiple dynamic changes in the illumination environment.

V. CONCLUSION

In this paper we have presented our efforts towards developing a system for rendering virtual objects in real time on everyday mobile devices. The novel contributions of our system are 1) that we capture and model the user‘s illumination in real-time and use it to light the virtual objects, making the objects appear to be situated in the real scene; and 2) the system is implemented in HTML and JavaScript and can therefore be accessed using standard web browsers on consumer-grade mobile devices through URLs without downloading or installing any software. We see great potential use of this system to enable widespread, realistic, interactive access to digital collections.

VI. REFERENCES

Blinn, James, and Martin Newell. 1976. "Texture and reflection in computer generated images." Communication. ACM.

Bodington, D, and J Thatte. 2015. "Rendering of Stereoscopic 360 ◦ Views from Spherical Image Pairs."

Currius, R, D Dolonius, U Assarsson, and E Sintorn. 2020. "Spherical Gaussian Light-field Textures for Fast Precomputed Global Illumination." Computer Graphics Forum.

Debevec, Paul. 2008. "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. Association for Computing Machinery. New York.

Ferwerda, James. 2014. "ImpastoR: A realistic surface display system." Vision Research.

Ferwerda, James, and Benjamin Darling. 2013. "Tangible Images: Bridging the Real and Virtual Worlds." In Proceedings of the 4th International Workshop on Computational Color Imaging. Berlin: Springer- Verlag.

Gardner, Marc-André, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. 2017. "Learning to predict indoor illumination from a single image." ACM Trans. Graph.

Iorns, Thomas, and Taehyun Rhee. 2016. "Real-Time Image Based Lighting for 360-Degree Panoramic Video." Image and Video Technology – PSIVT 2015 Workshops.

Liu, C, Z Li, S Quan, and Y Xu. 2020. "Lighting Estimation via Differentiable Screen-Space Rendering." IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops. Atlanta.

Brown, M and G Lowe 2007. "Automatic panoramic image stitching using invariant features." Int. J. Computer Vision 74.

Ng, Ren, Ravi Ramamoorthi, and Pat Hanrahan. 2003. "All- frequency shadows using non-linear wavelet lighting approximation." In ACM SIGGRAPH. Association for Computing Machinery. New York.

n.d. OpenCV.

https://docs.opencv.org/3.4/d5/d10/tutorial_js_root. html.

n.d. Three.js. https://threejs.org/.

Tien-Tsin, Wong, Heng Pheng-Ann, Or Siu-Hang, and Ng Wai-Yin. 1997. "Image-based Rendering with Controllable Illumination." Rendering Techniques.

Walton, D, D Thomas, A Steed, and A Sugimoto. 2017. "Synthesis of Environment Maps for Mixed Reality." IEEE International Symposium on Mixed and Augmented Reality.

WebGL. n.d. https://www.khronos.org/webgl/.

Yoo, J, and K Lee. 2008. "Real Time Light Source Estimation Using a Fish-Eye Lens with ND Filters." International Symposium on Ubiquitous Virtual Reality. Gwangju.