Deep Light Source Estimation for Mixed Reality

Bruno Augusto Dorta Marques[1], Rafael Rego Drumond[2], Cristina Nader Vasconcelos[1] and Esteban Clua[1]

[1]*Universidade Federal Fluminense, Instituto de Computação, Niterói, Brazil*
[2]*Universität Hildesheim, Institut für Informatik, Hildesheim, Germany*

Keywords: Mixed Reality, Deep Learning, Light Source Estimation.

Abstract: Mixed reality is the union of virtual and real elements in a single scene. In this composition, of real and virtual elements, perceptual discrepancies in the illumination of objects may occur. We call these discrepancies the illumination mismatch problem. Recovering the lighting information from a real scene is a difficult task. Usually, such task requires prior knowledge of the scene, such as the scene geometry and special measuring equipment. We present a deep learning based technique that estimates point light source position from a single color image. The estimated light source position is used to create a composite image containing both the real and virtual environments. The proposed technique allows the final composite image to have consistent illumination between the real and virtual worlds, effectively reducing the effects of the illumination mismatch in Mixed Reality applications.

1 INTRODUCTION

Recent advances on virtual reality platforms are allowing new paradigms of interaction to emerge. In particular, Head Mounted Displays (HMDs) have been developed by the industry to interface with video games and interactive simulations. The HMDs are responsible for increasing visual immersion and provide better user experience in the simulated environment.

However, the interaction between a user and simulated environment still relies on controllers or other unnatural hand devices such as the PlayStation Move, HTC Vive Controller, and Oculus Touch. These devices can break the user's immersion by not allowing a natural movement of the user's hands, *e.g.* the user is constrained to a limited range of movements and gestures due to the necessity of holding the controller in hand. To overcome this challenge less intrusive alternatives have been offered. These alternative devices seek to register the user's movements by a combination of sensors, such as accelerometers and gyroscopes, or by tracking devices such as RGB-D cameras that are able to detect body or hands movements (Kinect, Leap Motion) (Marin et al., 2014; Han et al., 2013; Zhang, 2012). However, the user is still represented in the virtual environment as an avatar, usually portrayed by a character that does not resemble the real user appearance. Furthermore, the user movements are usually exchanged for pre-made animation sequences that significantly differ from the user's movement. These problems can severely break the user immersion and impact the user experience in Augmented and Virtual Reality.

An alternative to the usage of avatars is to insert real footages of the user in the simulated environment. This approach solves the appearance and movement problem but introduces a new challenge, the lighting condition between the simulated environment and the user's real world conditions should match. When the lighting conditions differ, the resulting montage would introduce visual artifacts where the real footage is located and it becomes obvious to the user that his or her footages were inserted in an artificially created environment, we call this the illumination mismatch problem.

In this paper, we present a novel method to overcome the illumination mismatch problem. The Deep Illumination Estimation for Pervasive Systems is a method to estimate the illumination of the user's environment from a set of possible lighting configurations. Our method provides, for the virtual environment, high-level information of the lighting conditions in the user environment. This information can be used by the interactive simulation to adapt the lighting conditions in the virtual environment. Our method has the advantage of using only a single camera attached, or built, in the HMD device.

Marques, B., Drumond, R., Vasconcelos, C. and Clua, E.
Deep Light Source Estimation for Mixed Reality.
DOI: 10.5220/0006724303030311
In *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2018) - Volume 1: GRAPP*, pages 303-311
ISBN: 978-989-758-287-5

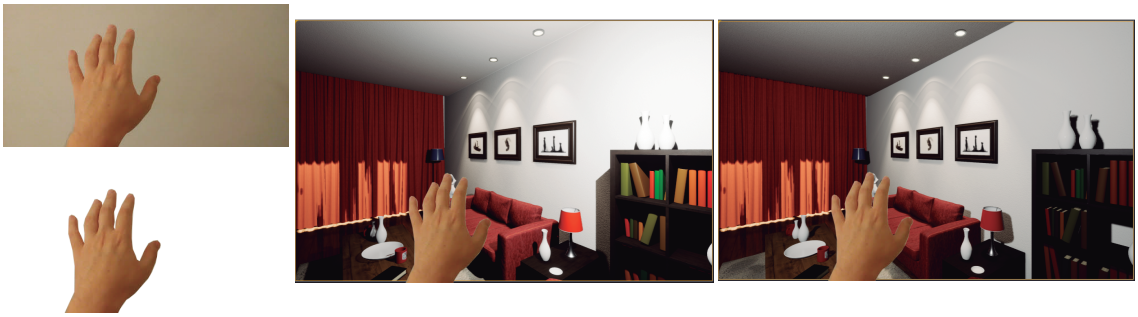

Figure 1: Top left: Input RGB image capturing real world illumination conditions. Bottom left: segmented hand image. Middle: overlay montage (real hand in a virtual scene). Right: resulting montage with adjusted light.

The hypothesis investigated in this work is whether the dominant light source position of a scene can be recovered from images of hands from the first-person point of view.

To archive this goal, we train a Convolution Neural Network (CNN) to classify the input RGB image to the corresponding lighting condition.

CNNs are used successfully for a wide range of problems involving classification, detection and segmentation of images. Recent CNN applications to solve such problems use images of different natures, including medical (Esteva et al., 2017; Kamnitsas et al., 2017), natural (He et al., 2016; Xie et al., 2016), synthetic images (Liu et al., 2016; Rajpura et al., 2017). It is not known to the authors any work that makes use of CNN for the recognition of illumination in an indoor scene.

To this task, we need a large data-set with annotated images for different scene illuminations. Since there is no such data-set available and acquire such dataset requires significant time effort, we created a synthetic data-set. We performed experiments to test if the CNN is capable of learning the lighting condition of acquired real images based on our synthetic data-set.

In Virtual Reality and Augmented Reality, the environment is seen from the users perspective. This first-person point of view implies that the most visible parts of the user's body are his or her hands. Thus, we focus on the hands of the user to retrieve the lighting information of the environment. We also consider that most of VR applications are used in an indoor environment, thus our method must work under this conditions.

The main contribution of this paper is the light source position estimation for the mixed reality that is capable of estimating illumination properties from a single RGB camera located in the HMD device. The method does not require any special hardware and can be implemented in any commercial HMD device. Furthermore, The system is used to generate a composition containing the user's hands and the virtual environment under the correct illumination.

This article is organized as follows: In Section 2 we describe the work related to the task of lighting recognition in real environments. In section 3 we give general aspects of the method and the application in augmented reality. The Sections 4 and 5 we detail the dataset construction and network architecture, respectively. In section 6 we detail the experiments and results found. The final conclusions of the paper are found in Section 7.

2 RELATED WORKS

Illumination estimation is important for different tasks, including image editing and scene reconstruction. Many aspects of lighting can be recovered including the visible spectrum of light, illuminating intensity, and position of light sources.

Different techniques for illumination estimation have been proposed based on probes and other intrusive objects (Calian et al., 2013), (Knecht et al., 2012), (Debevec et al., 2012), (Debevec, 2005) that must be inserted in the scene.

Calian *et al.* (Calian et al., 2013) created a 3D printed shading probe device that directly captures the shading of a scene. Positioning the device in the real scene, it was possible to achieve high-performance shading of virtual objects in the AR context.

Knecht *et al.* (Knecht et al., 2012) presented a rendering method for mixed reality systems that combines Instant Radiosity and Differential rendering. The environment light sources are approximated from the image of a fisheye lens camera that captures the surrounding illumination. Their method also requires the real scene geometry reconstruction, that is accomplished with the use of an RGB-D camera. Furthermore, a tracking device is required to estimate the pose of the camera.

Other methods require special equipment and

time-consuming processes, such as a fisheye lens and HDR camera setup to generate an environment map of the real environment (Pessoa et al., 2012).

These invasive methods hinder user immersion and can not be applied in all augmented reality scenarios. Our work does not rely on any intrusive device or previous setup step.

Similar to the purpose of our work, Boom *et al.* (Boom et al., 2015) proposed a method to estimate the light source position with an RGB-D camera. The method estimates a single light source position based on the geometry of the scene. They calculate the normals of the scene and perform a segmentation that finds regions with similar albedo in the original RGB image. They search for the light position that gives the best-reconstructed image by minimizing the distance between the reconstructed and real scene image. (Jiddi et al., 2016) addressed the problem for multiple point light sources based on the specular reflections in the scene. They also considered as input an RGB-D data provided by a sensor.

In the context of Mixed Reality Applications, (Mandl et al., 2017) estimates the illumination of the real ambient using physical objects on the scene as light probes. It is necessary to acquire the geometry of the light probe objects beforehand. The lighting is estimated by a 5 layers Convolutional Neural Network. They train multiple CNN's for each camera pose, resulting in plenty of trained CNN's. They use two different strategies, based on the camera pose, to select which CNN to be used in real time, interpolation and nearest neighbors selection. The CNN output a fourth order Spherical Harmonic constants that are used to create a Radiance Map of the scene. This radiance map is used to illuminate the virtual objects.

In our work, we train a single CNN to estimate lighting, this leads to advantages over (Mandl et al., 2017) work. Our CNN trains faster and we do not have to select which CNN to be used at run-time. The multiple CNN's in (Mandl et al., 2017) learns illumination using a single object as a light probe. For every new light probe, multiple CNN's must be trained. We have a single CNN that need to be trained only once and work in any application where the user's hands are visible.

Several methods have been developed for the illumination estimation of outdoor scenes (Hold-Geoffroy et al., 2016), (Lalonde et al., 2012), (Lalonde et al., 2010). These methods seek to estimate the parameters of a sky lighting model (Hošek-Wilkie (Hosek and Wilkie, 2012) model) that fits the environment illumination. Most of the methods infer parameters from shadow and shading cues, with the exception of the method described by (Hold-Geoffroy et al., 2016), that made use of a CNN to infer these parameters. Since these methods are focused on an outdoor lighting model it is not viable for indoor environments. The most common environment for playing games and virtual simulations are indoor environments. Thus, our work is focused on this kind of environment.

3 OVERVIEW

In the usual AR and VR setups, users are moving through the environment wearing an HMD device and interacting with their hands. Most of the time the user's hands and forearms are visible in the image captured by the built-in camera of the HMD device. In our scenario, we aim to produce a montage where those images containing the hands are inserted into the virtual environment. Our method is independent of the observed portion and may contain all or part of the user's hand, as well as the user's forearm.

Figure 2 shows the typical pipeline of our method. The system receives an image of the real environment, containing all or part of the user's hands. Since the CNN has been trained to classify an image containing only the user's hands, it is necessary to segment the image, isolating the user's hands in the image. This segmentation process is the second step of our algorithm, as seen in Figure 2. The segmented image is then supplied to a CNN, where classification occurs in one of the classes of illumination. This information is then provided to the Game Engine. The Game Engine adjusts lighting by changing the position of the main light source in the virtual environment.

The main light source of a scene can vary, it is determined by the scene designer of the simulation. The main light source for an indoor room can be represented by a single point light source. On the other hand. the main light source for an outdoor room can be accounted by a directional light representing the sun.

The last step is to create a montage using the segmented image and rendered virtual environment with the adjusted lighting setting. The montage can be created by overlapping the user's hands image with the virtual environment image. The HMD's camera is located approximately in the same position as the virtual camera on the simulated environment thus the user's hands share the same screen position. This pipeline must be run for each frame of the capturing camera.

Figure 3 represent the usual VR / AR setup, the objects that are, most of the time, in the camera's visible area are the user's hands and forearm. For the

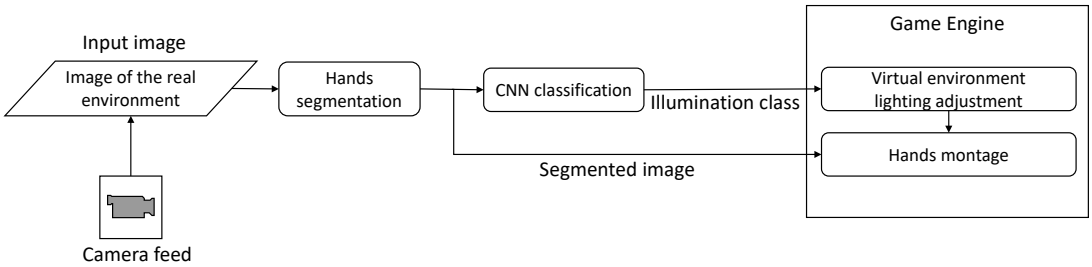

Figure 2: Lighting Estimation System Overview: typical usage: The input of the system is an RGB image containing one or two hands. The image is segmented to extract only the skin portion of the image (User's hands and forearms). A CNN estimates the 3D position of the main light source. The position is available to the Game Engine to adjust the virtual environment illumination.

indoor environment, a point light is used to indicate the main light source.

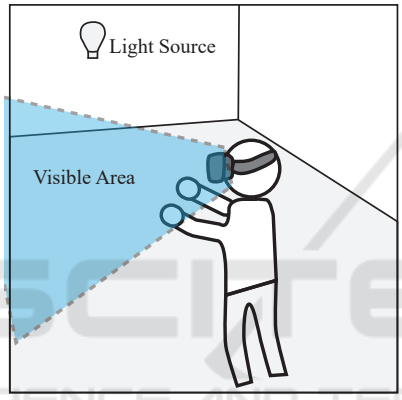

Figure 3: Usual VR and AR Setup. We consider the visible area as a view frustum with a horizontal field of view of 110 degrees and vertical field of view of 100 degrees.

With the goal of representing the mostly common environment for running mixed reality games and applications, we train a CNN to predict illumination conditions from a single indoor image containing a human hand. We use low dynamic range image obtained from a camera positioned in the HMD device. The camera captures the user hands illuminated by the real scene. We suppose that the scene appearance can be estimated from this image. The image is segmented to remove the background image and feed to a CNN that output a description of the lighting condition of the scene. This information can be used by the simulated environment to adapt the scene to the lighting condition and create a realistic montage of the simulated scene containing the virtual environment and the real user hands.

The description of the scene illumination contains a position of the dominant point light source. This 3D position is located on the surface of a sphere that was used in the creation of the database. To validate our approach, we overlay the segmented hand image in the virtual environment render where the dominant light source is indicated by the scene illumination description. A typical usage for the Lighting Estimation System is shown in the Figure 2, the process should be executed in real time, for every frame.

4 DATASET

To train the CNN it is necessary to use a dataset composed of images containing arms from a first-person viewpoint labeled with lighting conditions. Unfortunately, it is not known to the authors of this article that such dataset exists.

In order to have a labeled dataset, we constructed a synthetic dataset tailored for the light-estimation problem on Mixed Reality. The data-set consists of images containing human hands illuminated by different light sources. We rendered a pair of 3D modeled hands inserted in a black background. The hands consist of animated skeletal meshes performing one of the seven animations: Grab, Idle, Jump, Punch, Push, Sprint, Throw. There is a total of six hand models that were based on two different meshes (taking account female and male geometry) and six materials (used to simulate different skin colors).

The scene setup is illustrated in Figure 4. The hand is positioned in front of the camera with an offset distance of 50 cm. A single point light is used to simulate the light source. We initially positioned the light source in front of the camera with a distance of 200 cm. The 3D hands and the camera stay stationary. The point light source is movable.

To generate distinct lighting conditions, we need to change the position of the point light source. Since we are aiming to generate discretized lighting conditions representing the actual conditions of lighting in the user's environment, we sample evenly distributed points on the surface of a unit sphere. To accomplish

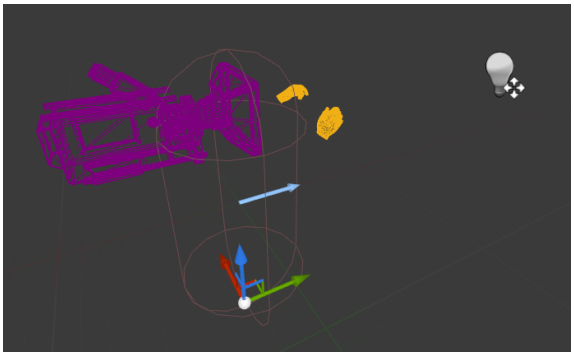

Figure 4: Scene Setup.

Algorithm 1: Lattice distribution algorithm.

procedure DISTRIBUTION(n) ▷ n is the number of samples
  $offset \leftarrow 2.0/n$
  $increment \leftarrow (3.0 - \sqrt{5.0}) * \pi$
  for $i = 0 \rightarrow n$ **do**
    $pointList.create()$ ▷ Create an empty list
    $y \leftarrow ((i * offset) - 1.0) + (offset/2.0)$
    $r \leftarrow \sqrt{1 - y^2}$
    $phi \leftarrow ((i+1) \bmod n) * increment$
    $x = \cos(phi) * r$
    $z = \sin(phi) * r$
    $pointList.add(x, y, z)$
  end for
  return *pointList*
end procedure

this goal, we choose the Fibonacci lattice distribution (González, 2010), (Marques et al., 2013) to generate n approximately evenly distributed points in a spherical region. These points represent the position of a valid point light of our dataset.

We also included a scenario with no direct incident light, where only an ambient light was used in the renderer. We used a screen-space subsurface scattering shader (Jimenez and Gutierrez, 2010) to realistically simulate the skin material.

All the images were processed by applying a motion blur to simulate the frames captured by the live camera in the real scene, the direction and amount of blurring are calculated according to the movement of the arms.

We also checked if at least one hand is visible in the final image; if no hand is visible, then we discard that image from the dataset. Later, we applied a centered crop in the images of the data-set for the purpose of keeping the aspect ratio of the images in the CNN training process. The resulting images are 512 pixels wide and 256 pixels tall. Example images can be seen in Figure 6.

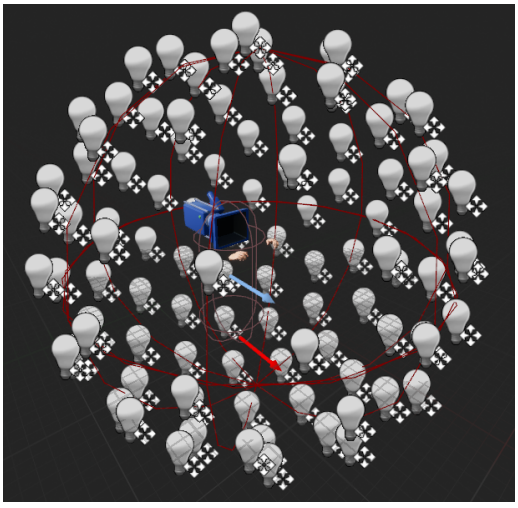

Figure 5: Light Distribution. The light sources are evenly distributed on the surface of a sphere.

Figure 6: Example images in the synthetic hand's illumination data-set. An image in this data-set contains a first-person view hands illuminated by a specific lighting setting.

We have created four variations of the dataset. The difference between them is the number of lighting settings: 5, 25, 50 and 100 light source position settings. The possible positions of the light source are uniformly distributed over the surface of the sphere using the Fibonacci Lattice distribution. The Fibonacci Lattice algorithm to generate n points is listed in Algorithm 1.

The synthetic hand's dataset, network model, and trained weights are publicly available at [Omitted due to blind review].

5 RESIDUAL NETWORK

Our method relies on image classification of lighting conditions by the observation of human hands. Deep Convolution Neural Networks represent the state of

the art methodology in several tasks related to visual content analysis, and the topology known as Residual Convolution Neural Network (ResNet) is the leading method among the existent CNNs in several challenges. Thus, our methodology is constructed by training a ResNet.

Residual networks are constructed based on *the deep residual learning* framework. Its network is constructed by replicating a basic building block (shown in Figure 7), that contains a set of convolution, nonlinear layers and a shortcut connection that skips one or more layers. The shortcut connections ease the training process and enable the usage of deeper networks without the degradation of the accuracy. The shortcut connection connects the input and the output of a building block.

In our work, we also use a bottleneck architecture to decrease the computational effort for the training process. The bottleneck convolution layer reduces the dimensionality of the input and recovers the original dimensions in the output. This is performed by two 1x1 convolution layers placed in the building block, the bottleneck layers are shown in Figure 7.

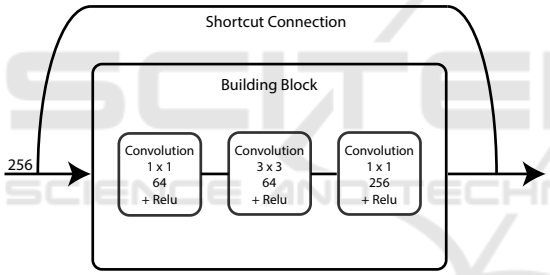

Figure 7: Basic Building Block for ResNet Network.

In our system, we use a 50-layer Residual Network. The construction of the network follows the same design rules as Resnet (He et al., 2016) architectures. Table 1 depicts the overall architecture of our residual network for each layer. The convolution layers have a kernel size of 3*x*3 pixels, with the exception of the first convolution layer. To create the 50 layer network, we repeat the building blocks based on the VGG and ResNet network (He et al., 2016; Simonyan and Zisserman, 2014). The repeat column of table 1 shows the number of repetitions for each building block. The blocks are created sequentially, the output of a previous block is connected to the input of the next one through a Non-Linear function called rectified linear units (ReLu). The ReLu Layer applies a simple non-linear function $f(x) = max(x,0)$ for every output in the previous layer.

6 EXPERIMENTS AND RESULTS

In order to measure the accuracy of the proposed method, we evaluated the performance of the CNN at predicting the main light position on the synthetic images dataset, so that the exact position of the light sources could be controlled. The model is trained on four variations of the Lighting Estimation Dataset, altering the discretization of the space considered as possible outputs of the network.

The first discretization has 5 lighting settings variation have 2943 training images and 1133 validation images. The final result is obtained on 451 test images. The second discretization has 25 lighting settings variation have 13496 training images, 5194 validation images, and 2071 test images. The third discretization considered has 50 lighting settings variation have 26676, 10269, 4099 images for training, validation and test respectively. Finally, the last discretization considered 100 lighting settings variation have 54471, 20965 and 8363 images for training, validation, and test. We evaluate top 1 accuracy.

Our CNN models were trained starting from pretrained weights obtained by training the same topology but for object classification in the ImageNet dataset (Deng et al., 2009) and Microsoft Coco (Lin et al., 2014). Such set of weights is public available (He et al., 2016). This is a well-known technique, named fine-tuning, adopted in order to reduce overfitting when the available dataset is small so that the network layers benefit from training in a larger one.

For the 100 lighting settings variation, our network learned 27,657,316 parameters and took a processing time of about 10 hours.

The network variation for 50, 25 and 5 lighting settings learned 25.609.266, 24.585.241 and 23.541.231 parameters and took a processing time of 5:01, 3:42, and 00:19 hours, respectively, for the training process in the GPU.

All of the tests were performed in a machine with the following specification: I7 4790 @ 3.6Ghz. 24 GB Ram. Nvidia Geforce Titan X.

The input of the network is of the same size as the images on the data-set, thus having a size of 512 x 256 pixels.

The inference time for a single image is about 0.15 seconds, executed on the CPU.

The CNN output a probability distribution over the 5/25/50/100 possible classes (lighting settings) for each prediction. The top-1 accuracy is the accuracy considering only the CNN output class with higher probability.

The accuracy results can be seen on Table 2. Increasing the number of lighting settings implies a

Table 1: Network Architecture.

Block	Kernel Size	Stride	Pad	Output	Repeat
Convolution 1	7 x 7 Convolution 3 x 3 Max Pooling	2 2	3 0	64	1
Convolution 2	1 x 1 Convolution 3 x 3 Convolution 1 x 1 Convolution	1 1 1	0 1 0	64 64 256	3
Convolution 3	1 x 1 Convolution 3 x 3 Convolution 1 x 1 Convolution	1 1 1	0 1 0	128 128 512	4
Convolution 4	1 x 1 Convolution 3 x 3 Convolution 1 x 1 Convolution	1 1 1	0 1 0	256 256 1024	6
Convolution 5	1 x 1 Convolution 3 x 3 Convolution 1 x 1 Convolution	1 1 1	0 1 0	512 512 2048	3
Average Pooling	7 x 7 Avg. Pooling	1	0	2048	1
Fully Connected + Softmax	Fully Connected layer + Soft Max	-	-	50	1

Table 2: CNN Accuracy for lighting estimation.

Lighting settings	Top-1 accuracy	Training Time (hours)
5	93.87 %	00:19
25	83.15 %	03:42
50	82.73 %	05:01
100	81.51 %	10:16

Figure 8: On the left the input images. In between montage image without lighting adjustment. On the right montage image with lighting adjustment.

harder problem, thus decreasing the accuracy of our model. For the 100 lighting settings dataset, we still accomplish a top 1 accuracy of 81.51%.

As we increase the number of lighting settings, the difference between the estimated position decreases, leading to more subtle differences in the final image, as can be seen in Figure 9. While there is a big difference between the classification with 5 and 25 lighting settings, the classification with 50 and 100 lighting settings generates subtle differences that are hard to be identified by the user.

To further evaluate the quality of our system, we performed a test with the full pipeline in Figure 8. The input images are displayed on the left side of the figure. The input image is a human hand in an arbitrary position. The input image has been processed in order to remove all content that does not belong to the user's hands. In the middle column of image 8 we show the environment in a random lighting setting. on the right column, we show the final assembly where the environment has its adjusted lighting configuration and the overlap of the input image with the virtual environment. These results were obtained using the trained network for 100 lighting classes.

7 CONCLUSION

We presented a point light source position estimation system for mixed reality which is able to estimate the light source position of a 3D scene. Different from previous works, our system uses only a low dynamic range camera and do not requires any additional hardware or user intervention. The system is suitable for mixed, virtual and augmented reality applications and operates at interactive rates. We evaluate the performance of our novel system on different user scenarios.

As future works, the proposed methodology can be extended to retrieve descriptors of the illumination chrominance and intensity, as well other illumination parameters that can affect the illumination perception realism.

Figure 9: CNN Lighting estimation with a different number of lighting settings. The top image is the scene illuminated by the 100 lightings settings CNN estimation. From left to right: 100 lighting settings, 50 lighting settings, 25 lighting settings and 5 lighting settings. Each column shows the color image and the difference image between the corresponding lighting setting and the 100 lighting settings image.

ACKNOWLEDGEMENTS

The authors thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for the financial support of this work and Nvidia for providing GPUs.

REFERENCES

Boom, B. J., Orts-Escolano, S., Ning, X. X., McDonagh, S., Sandilands, P., and Fisher, R. B. (2015). Interactive light source position estimation for augmented reality with an rgb-d camera. *Computer Animation and Virtual Worlds*.

Calian, D. A., Mitchell, K., Nowrouzezahrai, D., and Kautz, J. (2013). The shading probe: Fast appearance acquisition for mobile ar. In *SIGGRAPH Asia 2013 Technical Briefs*, page 20. ACM.

Debevec, P. (2005). Image-based lighting. In *ACM SIGGRAPH 2005 Courses*, page 3. ACM.

Debevec, P., Graham, P., Busch, J., and Bolas, M. (2012). A single-shot light probe. In *ACM SIGGRAPH 2012 Talks*, page 10. ACM.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118.

González, Á. (2010). Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical Geosciences*, 42(1):49–64.

Han, J., Shao, L., Xu, D., and Shotton, J. (2013). Enhanced computer vision with microsoft kinect sensor: A review. *IEEE transactions on cybernetics*, 43(5):1318–1334.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hold-Geoffroy, Y., Sunkavalli, K., Hadap, S., Gambaretto, E., and Lalonde, J.-F. (2016). Deep outdoor illumination estimation. *arXiv preprint arXiv:1611.06403*.

Hosek, L. and Wilkie, A. (2012). An analytic model for full spectral sky-dome radiance. *ACM Transactions on Graphics (TOG)*, 31(4):95.

Jiddi, S., Robert, P., and Marchand, E. (2016). Reflectance and illumination estimation for realistic augmentations of real scenes. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'16 (poster session)*.

Jimenez, J. and Gutierrez, D. (2010). *GPU Pro: Advanced Rendering Techniques*, chapter Screen-Space Subsurface Scattering, pages 335–351. AK Peters Ltd.

Kamnitsas, K., Ledig, C., Newcombe, V. F., Simpson, J. P., Kane, A. D., Menon, D. K., Rueckert, D., and Glocker, B. (2017). Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical image analysis*, 36:61–78.

Knecht, M., Traxler, C., Mattausch, O., and Wimmer, M. (2012). Reciprocal shading for mixed reality. *Computers & Graphics*, 36(7):846–856.

Lalonde, J.-F., Efros, A. A., and Narasimhan, S. G. (2012). Estimating the natural illumination conditions from a single outdoor image. *International Journal of Computer Vision*, 98(2):123–145.

Lalonde, J.-F., Narasimhan, S. G., and Efros, A. A. (2010). What do the sun and the sky tell us about the camera? *International Journal of Computer Vision*, 88(1):24–51.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Liu, X., Liang, W., Wang, Y., Li, S., and Pei, M. (2016). 3d head pose estimation with convolutional neural network trained on synthetic images. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 1289–1293. IEEE.

Mandl, D., Yi, K. M., Mohr, P., Roth, P., Fua, P., Lepetit, V., Schmalstieg, D., and Kalkofen, D. (2017). Learning lightprobes for mixed reality illumination. In *16th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, number EPFL-CONF-229470.

Marin, G., Dominio, F., and Zanuttigh, P. (2014). Hand gesture recognition with leap motion and kinect devices. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 1565–1569. IEEE.

Marques, R., Bouville, C., Ribardière, M., Santos, L. P., and Bouatouch, K. (2013). Spherical fibonacci point sets for illumination integrals. In *Computer Graphics Forum*, volume 32, pages 134–143. Wiley Online Library.

Pessoa, S. A., Moura, G. d. S., Lima, J. P. S. d. M., Teichrieb, V., and Kelner, J. (2012). Rpr-sors: Real-time photorealistic rendering of synthetic objects into real scenes. *Computers & Graphics*, 36(2):50–69.

Rajpura, P. S., Hegde, R. S., and Bojinov, H. (2017). Object detection using deep cnns trained on synthetic images. *arXiv preprint arXiv:1706.06782*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2016). Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*.

Zhang, Z. (2012). Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10.