Final Project Proposal

Project Title: Scene Lighting Estimation From 2D Images
Link to Git repo for project: https://github.com/AlstonXiao/IlluminationDetection
Team member[s]:

- Name: Yan Xiao
- Email: yxiao65@wisc.edu

- Name: Collin Lenz
- Email: ctlenz@wisc.edu

- Name: Chris Chu
- Email: cchu34@wisc.edu

Problem Statement:

- Illumination estimation from 2D images/videos can be extremely difficult or inaccurate without encompassing panorama shots or external equipment such as light probes. This becomes important for MR/AR, where you need to insert virtual objects into any 3D location in a 2D scene while maintaining the appearance of realistic lighting.

Motivation/Rationale:

- A significant amount of AR applications and functions popular in the world today utilize "regular" 2D RGB image/video input with limited fields of view, such as phone or tablet cameras. Illumination detection techniques on these types of images are necessary when attempting to simulate realistic lighting for inserted virtual objects without the need for external equipment or advanced camera systems.
- Generally, we find the VR/MR/AR continuum quite interesting, specifically game development and filming, so the implications/applications of illumination estimation piqued our interest.

Current state-of-the-art:

- Current state-of-the-art illumination detection and virtual insertion models like Lighthouse and Neural Illumination (referenced below) create near-photorealistic lighting using specifically designed convolutional neural networks to optimize pitfalls of previous models. Neural Illumination breaks up their model into three sub-modules to address the complexities of previous models. Lighthouse claims that while Neural Illumination improved upon persistent problems, it led to "spatial incoherence" due to information being separated into three separate modules in the CNN. Their solution was to construct a 3D CNN to resolve previous complexities while also managing "spatial incoherence" from Neural Illumination's model.

Explain how you contemplate going about it:

- Read through, analyze, and understand our current reference papers.
- Decide what the best way of representing lights in the AR/MR application is. This representation should be compact, easy to use in render engines, and quick in the calculation.
- Find, label, and pre-process the data set we find on the web.

- Determine our CNN architecture, activation function, loss function, parameters, training set, training method, and training dataset.
- Choose our coding language and see what libraries are available and useful for the program.

Why do you think existing approaches cannot adequately solve this problem? Why do you think your solution will work better?

- The current state-of-the-art proposed a great solution to estimate the environment lights. Still, they are not suitable for real-time AR/MR applications. First, the time they need to generate the lightmap is long, compared to 16ms or less per frame in AR/MR applications. The full-size environment map will also affect memory performance. Secondly, these applications are mostly trained on discrete images, while AR/MR cameras will produce continuous images.
- Our approach will lean more towards speed than realism by reducing several features. We want to create a "good enough" result within the speed limit.
- We will focus on the result consistency between pictures we take at a different angle in the same lighting setup. We want the AR/MR cameras to generate the same lightmaps when they are moved around.

What we intend to accomplish:

- The result would be a software that the user can feed images into as input and receive the estimated light position as output. Then the user can use this data to shade the objects in the virtual world.
- We intend to take real-world photos and use our software to generate estimated scenes' lighting data. Then we will use this data to shade a virtual object and merge it with the original image.
- We will compare the results between a real object in the scene and a virtual object, shaded in Unity.
- The performance that we will evaluate is the quality of the light position of a virtual object at all angles in comparison to the quality of the light position of the object in reality. The closer to reality an AR object is shaded, the higher evaluation the object will receive. We can use a CNN and an evaluation set to compare the shadings of the object together and run a correlation similarity test.

Timeline:

- Oct 9: Finish reading paper and gather initial ideas
- Oct 20: Determine what type of light representation/result should look and what kind of pre-processing. Prepare the input data to the network and label them.
- Nov 1: Start the first attempt to run the network and build the software wrapper for the final representation
- Nov 20: Should have a working/reasonable result at this point.
- Dec 1: Finish the training, start working on the final presentation and website.

References:

https://arxiv.org/pdf/1906.07370.pdf

https://openaccess.thecvf.com/content_CVPR_2020/papers/Srinivasan_Lighthouse_Predicting_Lighting_Volumes_for_Spatially-Coherent_Illumination_CVPR_2020_paper.pdf

https://openaccess.thecvf.com/content_ICCV_2019/papers/Gardner_Deep_Parametric_Indoor_Lighting_Estimation_ICCV_2019_paper.pdf

https://openaccess.thecvf.com/content_cvpr_2017/papers/Hold-Geoffroy_Deep_Outdoor_Illumination_CVPR_2017_paper.pdf