Visual Geometry and Modelling
Istituto Italiano di Tecnologia

Pattern Analysis and Computer Vision
Istituto Italiano di Tecnologia

UNIVERSITÀ DEGLI STUDI
DI GENOVA

VISUAL GEOMETRY AND MODELLING (VGM)
PATTERN ANALYSIS AND COMPUTER VISION (PAVIS)
DEPARTMENT OF ELECTRICAL, ELECTRONIC AND TELECOMMUNICATIONS ENGINEERING AND NAVAL ARCHITECTURE (DITEN)

Ph.D. in Science and Technology for Electronic and Telecommunication Engineering

Curriculum: Computational Vision, Automatic Recognition and Learning

---

Leveraging Multi-View Information for Scene Understanding

Mohamed Dahy Abdelzaher Elkhouly

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Supervisor: Dr. Alessio Del Bue
Co-Supervisor: Dr. Stuart James

Coordinator of the PhD Course: Prof. Mario Marchese

FEBRUARY 2021 - XXXIII CYCLE

Acknowledgements

I had the pleasure to share these three years of Ph.D. with a lot of kind, brilliant, and inspiring people.

The first words of deep gratitude is to my supervisor, Alessio Del Bue, for the opportunity to learn from him, the guidance, the inspiration, and the mentorship. To my co-supervisor Stuart James, for all the brilliant insights on 3D modelling and geometry, all the good ideas and experiments that make some of my favorite parts in this document, that make my successful entry in this domain. The accomplishment of this work would not be possible without their mentorship and continuous encouragement throughout this period. I can not thank them enough, but the saying of the prophet muhammad (pease be upon him), who said: **"He who follows a path in quest of knowledge, Allah will make the path of Jannah easy to him. The angels lower their wings over the seeker of knowledge, being pleased with what he does. The inhabitants of the heavens and the earth and even the fish in the depth of the oceans seek forgiveness for him. The superiority of the learned man over the devout worshipper is like that of the full moon to the rest of the stars (i.e., in brightness). The learned are the heirs of the Prophets who bequeath neither dinar nor dirham but only that of knowledge; and he who acquires it, has in fact acquired an abundant portion."** can express what I want to say. Thanks to our new team member Theodore Tsesmeils who left his print in our work.

I'm fortunate to have crossed paths with many people that made my stay in Genova a beautiful experience. VGM, PAVIS, IIT, and Unige are incredible places to be as a young researcher. I'm grateful to Dr. Alessio Del Bue, Prof. Mario Marchese, and all staff for the helpful, cooperative and lively environment. Special thanks to my batch fellows Muhammad Shahid, Muhammad Abubakar Yamin, Matteo Bustreo, Danilo Greco for sharing some pleasing moments of life. It is my pleasure to be in the same institution with Pietro Morerio and Jacopo Cavazza to learn from their

insights about deep learning. I am also thankful to human resource staff at IIT and Unige for their support. Thank you to all my friends from IIT. I Kept this special thanks with respect to the first person I have met from IIT Dr. Diego Sona.

I am most grateful to my family (My mother, father, sister, brother, and all of my Pakistani brothers in IIT). They are the most inspiring examples I have. Especially my parents for teaching me good moral values which are more important than any kind of science. Thanks for their endurance, inspiration, and sacrifices for the whole period of my studies. I can not express my feelings and appreciation toward them, but at least I can say as Allah told us to say: "**My Lord, have mercy upon them as they brought me up [when I was] small.**"(17:24). To my favorite person in the world, Sara, my wife, for the unique care, understanding, closeness, for all the love, patience, and support, I will also mention what Allah said:**And (it is one) of His signs that He has created spouses for you from your own species that you may find comfort in them. And He has induced mutual love and tenderness between you. Behold! there are signs in this for a people who would reflect.**(30:21).

Abstract

The Humans can effortlessly recognize and perceive the information of the 3D scene on their daily pass of dozens of scenes, e.g. office, bedroom, elevator, kitchen. Even though this process is effortless for humans, it is an open challenge in the Computer Vision domain since the field was first established 60 years ago. 3D scene understanding from multi-views is key for the success of applications such as robot navigation and autonomous driving, etc. In this thesis, we seek to exploit multi-view scenes information to help indoor scene understanding to overcome challenges dependant on visual effects like shadows, specularities, and occlusions towards better scene understanding. Towards this goal, we propose techniques based on multi-view scenes with corresponding 3D geometry to estimate semantic color-names, detect multi-view specularities, estimate multi-view 3D mesh colors, and estimate light source positions. We use available large-scale datasets (e.g. Matterport3D) for annotating real-world images datasets like Matterport3D Color Naming Dataset (M3DCN), and Matterport3D Light Sources dataset (M3DLS) while generating and rendering new 3D synthetic datasets like LIGHT Specularity Dataset (LIGHTS) to serve as evaluation and analysis datasets for semantic color-naming, light source position estimation, and high-light specularity detection problems respectively. We demonstrate the effectiveness of our proposed techniques in comparison to the state-of-the-art techniques and show significant improvements over the baselines in quantitative and qualitative evaluations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In Computer Vision, multi-view scene analysis has been extensively studied and gained significant interest within several tasks and various practical applications, like 3D shape estimation [16, 17], object recognition [18, 19, 20], categorization [21], modifying correspondences variations [22], person and vehicle re-identification [23, 24, 25], reconstruction [26, 27], and retrieval [28]. There is no doubt that these tasks are heavily dependent on visual features , which means that they are prone to visual errors in the presence of light-related effects e.g. specularities, shadows, and colored-lights. These effects can lead to a false decision, e.g. for colored-lights, a white-colored object in red-colored light appears as red (see Figure 1.1). As for specularities, black objects in the presence of specularities can appear partially or totally in the same color of the reflected light-source based on their geometrical shape, surface properties, and point of view. As for shadows, a light-blue object in the presence of shadows can appear as a dark-blue object, while in the same circumstances white will be visible as gray, while cast shadows from detailed objects like trees on another object cause to many edges to the other object and it will take the shape of the tree leaves, which for sure can mislead edge-based techniques. All of these different examples were based on the main elements of vision, which is light and reflecting objects, and were only affecting one visible element which is the color that affects each task from the earlier mentioned tasks. These effects change

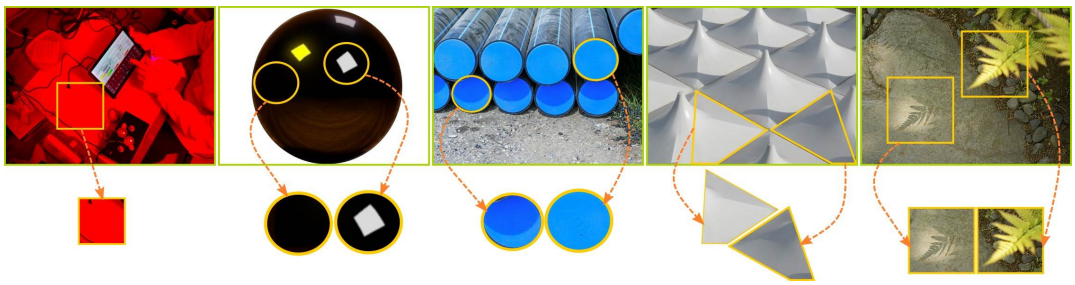


Figure 1.1: Examples of light-related effects on objects, from left to right white table under red-colored light, a black object under three light sources, blue-end pipes under light and shadow, a white umbrella under light and shadow, and finally the shadow of tree leaves.

objects colors partially or totally, control edges, form shapes, and with no doubt, it can control the perceived information which could mislead scene understanding with false interpretations. Compared to a single view scene, a multi-view scene provides a wealth of interpretations of the scene and its objects such as structure, and location info, and can overcome difficulties created by factors such as occlusion and distortion of perspective that are defective in the single image scenes. Moreover, it also presents relationships between objects in different images particularly for rigid objects, sequential relationships for data from time series, semantic relationships for representative object points, which are absent as part of object information and scene interpretation in the single view scenes. Despite this, it is a challenging task due to variations from multi-view especially the amount of captured light, different camera tuning parameters, specularities, and occlusions. For example, in Figure 1.2 same part of the vase (green rectangle) captured with different brightness from different views. The same effect is noticeable on the book (orange rectangle) while the page colors on the second view are darker than the colors on the first view. This effect is due to different camera tuning parameters and the distance between camera and object, it results in inconsistencies in obtaining colors for the 3D mesh as shown in the ceiling and wall colors of the 3D reconstructed mesh. In different view-

points High-light specularity effect varying in covering some spots on the red bowl (blue rectangle), Also, Objects are occluding parts of each other e.g. Vase and knife holder. In this thesis, we leverage the multi-view information for obtaining impor-

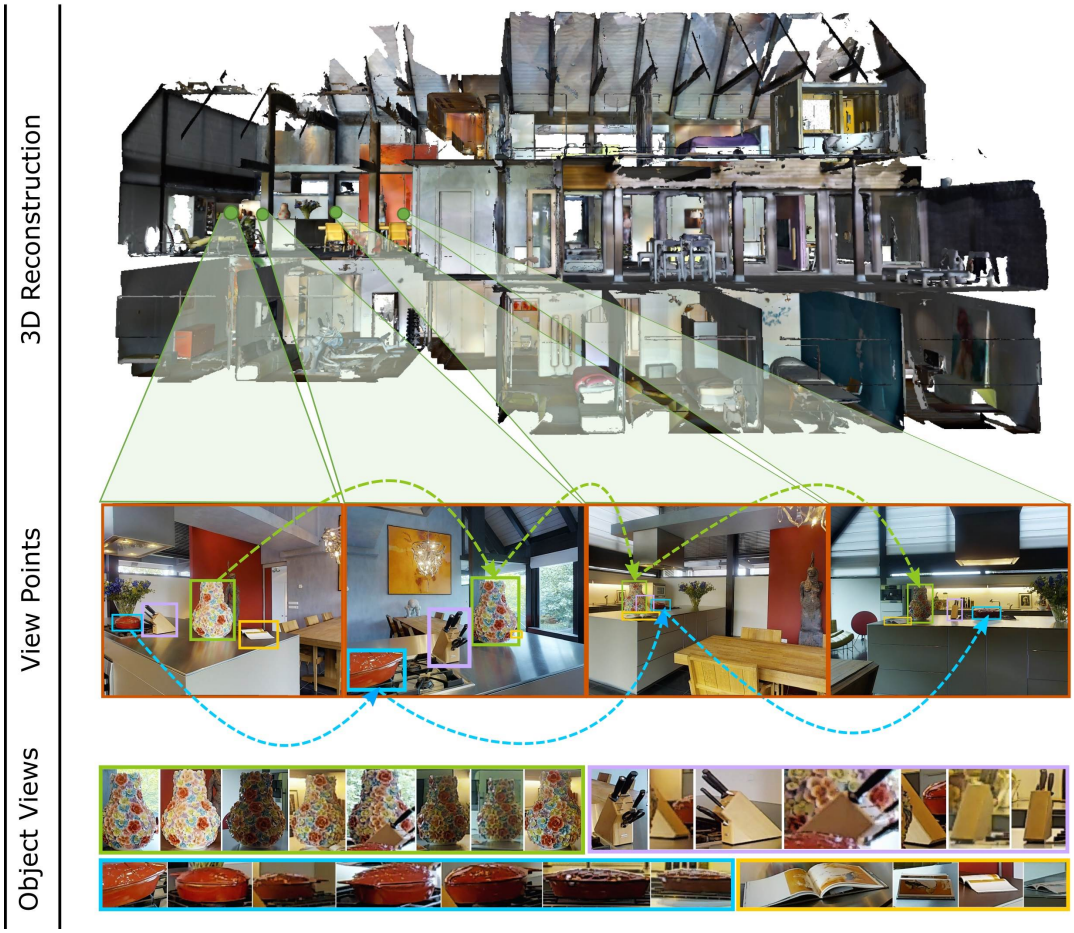


Figure 1.2: Multi-view images of a static scene from different view-points have variations in objects perceived amount of light, occlusions, and specularities based on view-point location and angle concerning objects and light sources, while at the same time have rich information to overcome difficulties faced in individual images.

tant information that can be used as direct or complementary information in scene understanding to avoid these aforementioned visual errors. Specifically, we focus on detecting multi-view specularities, estimating 3D mesh colors from multi-view images, and classifying multi-view objects colors into their corresponding semantic color-names with the analysis of the shadows effect.

1.1 Motivation

The analysis of multi-view scenes is very promising and does indeed provide information that is useful especially for computer vision techniques in many domains, but there are various limitations that have led to interesting research avenues and motivated our work in this thesis. The principal motivation of the work discussed here is to overcome these outstanding issues and propose solutions for them. We proposed solutions for four main issues: semantic color naming estimation of rigid static objects from multi-view correspondences, the extension of the method with the detection and removal of shadows, highlight-specular detection from multi-view images, estimating consistent colors for a 3D mesh from multi-view images, and finally estimating light source position. A large body of work in Computer Vision fields such as object recognition [18, 19, 20], reconstruction [26, 27], or retrieval [28] directed to dependency on visual features such as edges, corners, contours, etc. represented as organizations of color values in pixels, in other words, they are shape-dependent. While in this thesis we are seeking other kinds of information, which is based on light and surface properties and variate with it, even if the shape and structure are fixed. We believe that directing research toward light-scene interaction analysis would be beneficial to gain wealthy information to solve outstanding issues. We also believe that our obtained multi-view information will not only be important for multi-view domain only but also reaching for single-view tasks that require big annotated data for training. In the future, multi-view algorithms can extract object features from multi-view images, then reflect these estimations for each object on its single-view instances, which can be used as training data for big-data trainable systems.

1.2 Thesis Objective

The overall aim of our work in this thesis is to advance research for leveraging multi-view images to extract information that would help in scenes understanding of large-scale multi-view visual data specifically focusing on light-dependent or affected visual elements. Specifically, we want to exploit multi-view images for:

- Estimating objects color-names, which serves in semantics-based computer vision tasks such as Visual Question Answering (VQA), and Automatic Image Captioning (AIC), and analyzing the effect of shadows information on the estimation;
- Detecting multi-view light-related effects specifically specular high-lights;
- Estimating consistent colors for 3D mesh elements from its correspondences in multi-view images;
- Exploiting specularities and consistent mesh color estimations to infer light source position.

1.3 Contributions

In this thesis, we explore the idea of using multi-view images collections to infer rich and precise information of 3D scenes through computer vision and statistical techniques to provide enhanced higher performance techniques in comparison to state-of-art related techniques for overcoming existing visual outliers. In particular, we made the following contributions:

- Propose three simple and robust aggregation techniques for multi-view color naming that utilizes wisely the available information across views [29].
- Study the effect of shadows in multi-view color labeling and propose to integrate shadow information in the aggregation algorithms to enhance the performance of color naming.
- Present the first multi-view 3D dataset for color naming, Matterport 3D Color Naming dataset (M3DCN).
- Present a new physically based rendering dataset (LIGHTS) for light analysis from high-quality models with detailed light modeling.
- Present a framework for multi-view extension from single-view approaches for specular highlight detection that Exploits the benefits of single-view highlight specular detection techniques and the scene geometry to detect the specularities in each view.
- Estimate consistent per-face colors for multi-view reconstructed mesh.
- Refinement method for prior mesh textures by correcting the textures using estimated per-face colors.

- Present the annotated Matterport3D Light Sources dataset (M3DLS) for light source evaluation.

1.4 Overview of the Thesis

The rest of the thesis is presented as follows.

Chapter 2 :

This chapter presents some background for the research presented in this thesis. We start with presenting how do humans see the world views, and how computers convert these views into images and then presenting the digital images. We also present the color Spaces, light modeling, shading, and rendering.

Chapter 3 :

Reviews the state-of-the-art methods related to multi-view scene understanding problems in three different problems starting from the related work of color-naming problems in a single view and the available shadow detection and removal techniques. Whereas for highlight-specularities detection we present the state-of-the-art methods for detecting it in both single and multi-view cases and listing the available vision datasets which could be valid for light analysis that could be used for validating such light-related problems. We also perform a detailed analysis of previous research works that managed to estimate the colors of the 3D mesh. Finally, we focus on and investigate light source position estimation state-of-art methods.

Chapter 4 :

Introduces a robust approach for color-name estimation for multi-view reconstructed scenes. It provides threes methods based on probabilistic latent semantic analysis and statistical methods for feature aggregation from multi-view. We present our

new dataset for multi-view color-name evaluation purposes. A detailed analytical study About different forms of input data and the effect of correcting or removing undesired outliers (shadow in our study) is presented.

Chapter 5 :

Proposes a method to detect specular highlights in multi-view scenes by analyzing its behavior in different views from different viewpoints with different angles. Motivated by its effect on multi-view images, we present a fast and efficient algorithm to detect specularities in 3D scenes based on mesh faces. In this chapter, we present a new light source evaluation dataset and our proposed evaluation metric for specularities related problem. In the end, we present a quantitative and qualitative study to express the efficiency of our proposed technique versus state-of-the-art technique.

Chapter 6 :

Introduces a novel method for estimating colors of 3D reconstructed scenes from multi-view images. The proposed method processes and correlate colors values from all frames together based on overlapped faces between them. We rely on the cross-correlation relation between overlapped frames to estimate the mesh colors. We demonstrate the consistency of our estimated colors through comparison with state-of-the-art mesh coloring or texturing techniques. Moreover, we show how we can use our estimated colors to correct the results from other texturing techniques to provide consistent color high resolution textured mesh.

Chapter 7 :

Presenting an extension of our work to localize the light source in real multi-view reconstructed scenes. We show our preliminary proposed system and the details of its stages to detect the light source position. Preliminary discussion on the estimations is presented including shortcomings of our research work and further considerations.

Chapter 8 :

Presents the overall summary of our contribution to multi-view high-level scene understanding, the shortcoming of our research work, and further ramifications and extension as a solution to the challenges.

Chapter 2

Background

Humans took a while to understand light. While The Greeks developed a theory of vision in which rays of light emanating from the eyes to objects, and by which we can see those objects. On the other side, Arab scientist, Ibn al-Haytham (from 965 to 1039), said that we can see the objects because of the sun's rays of light; streams of tiny particles traveling in straight lines hit the objects and reflect from them into our eyes, forming images (see Fig2.1). Now, regarding how do these images captured in computers, this is the field of Digital imaging or digital image acquisition which is the creation of a representation of the visual characteristics of an object according to the Federal Agencies Digital Guidelines Initiative Glossary (FADGI). These characteristics could be a physical scene or the interior structure of an object. In all classes of digital imaging, the information is converted from rays (e.g. light rays, x-rays, or others) reflected from objects by image sensors into digital signals that are processed by a computer and made output as a visible-light image. These images are composed of picture elements instead of reflected rays which are known as pixels. each with finite, discrete quantities of numeric representation for its intensity. These pixels could be colored or gray level or even binary (black and white) based on the acquisition sensors. These sensors could be organized into a spatial two-dimensional array which denotes the x, and y spatial coordinates on the x-axis and y-axis, respectively. Digital images do not only including the familiar

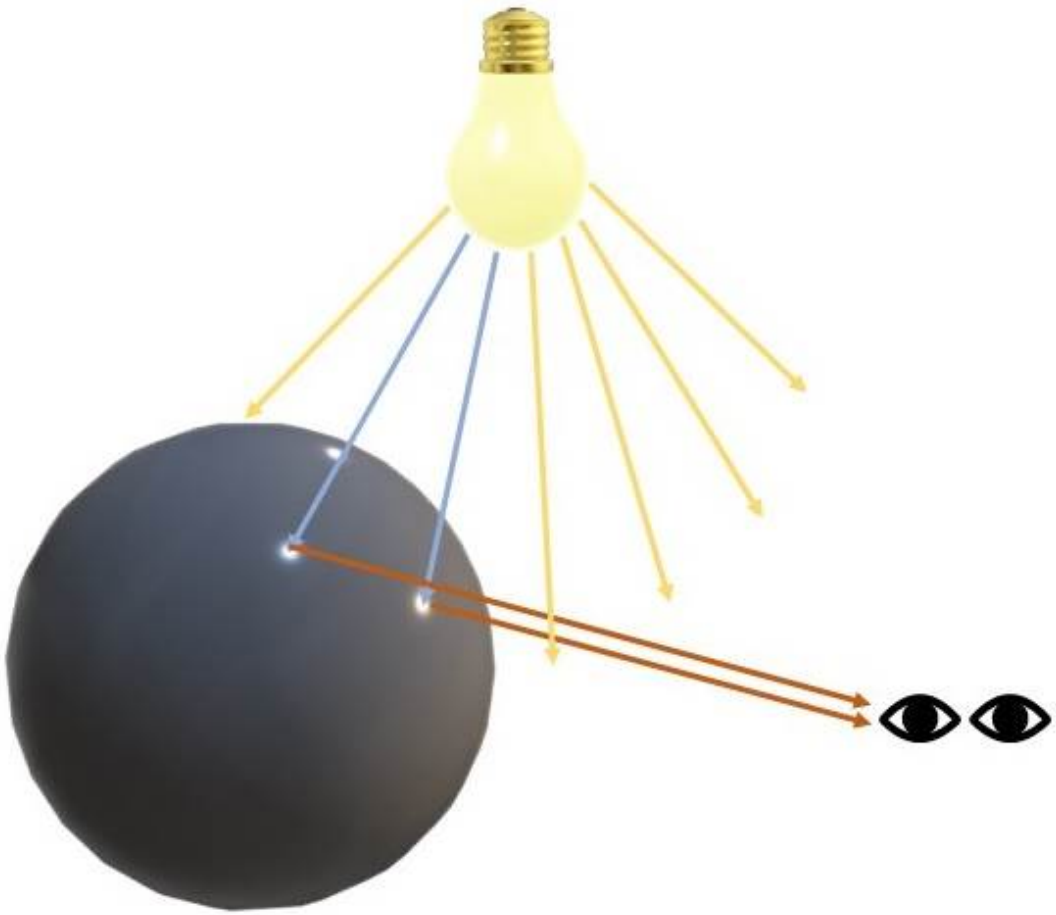


Figure 2.1: Light rays traveling from the light source to object then reflected to the human eye allowing the eye to see the object.

images captured by light cameras like mobile phone cameras, but also including digital X-ray imaging such as digital radiography, fluoroscopy, and CT which is dependant on sensors specified for capturing x-rays instead of light rays and so for other types of digital images such as Magnetic resonance imaging (MRI) which is a medical imaging technique used in radiology which using MRI scanners to capture radio waves, and digital gamma-ray imaging such as digital scintigraphy, SPECT, and PET. Sound also allows ultrasonography (such as medical ultrasonography) and sonar and recently acoustic images. In our thesis, we are concerned about digital images captured based on light reflections.

2.1 Color Space

The color of an object or the color of light can be seen as the result of many different light colors from the visible spectrum mixed together. The problem with this representation is that human eyes can not directly perceive spectral power distribution. Therefore, to represent colors, we need to convert spectral power distributions into another representation that is better suited for human vision. Such representations are called color models or color spaces. One of the most famous color models is the CIE XYZ color model which is the foundation of all color models as well as the RGB model which is a popular model in computer graphics. In short words, we can see a color model as a technique of mapping the values from spectral power distributions to three values which can more directly relate to the way human eyes perceive colors. More useful resources regarding this topic could be found in [30]. A color model could be an abstract mathematical model describing the way colors can be represented as tuples of numbers (e.g. triples in RGB or quadruples in CMYK), or a color model with no associated mapping function to an absolute color space which is called arbitrary color system with no connection to any globally understood system of color interpretation. For example, Adobe RGB and sRGB are two different absolute color spaces, both based on the RGB color model. When defining a color space, the usual reference standard is the CIELAB or CIEXYZ color spaces, which were specifically designed to encompass all colors the average human can see. As the color space is the model that can be used to represent as many colors as our visual system can possibly perceive, so the colors are usually referred to as the colors that the human visual system can perceive. The range of possible colors that can be represented by a particular system is called the **gamut**. A color space in its most simplistic form can be seen as the combination of three primary colors (red, green

blue) in a color additive system as shown in Fig 2.2. Adding any of these colors to each other can produce a wide range of colors (this is the base of the RGB color model). The gamut of the CIE xyz color space, for example, has a horseshoe shape as shown in Fig 2.2, and so this shape varies depending on the color space used. This thesis and previous related works mainly depending on one or more color spaces on the proposed work.

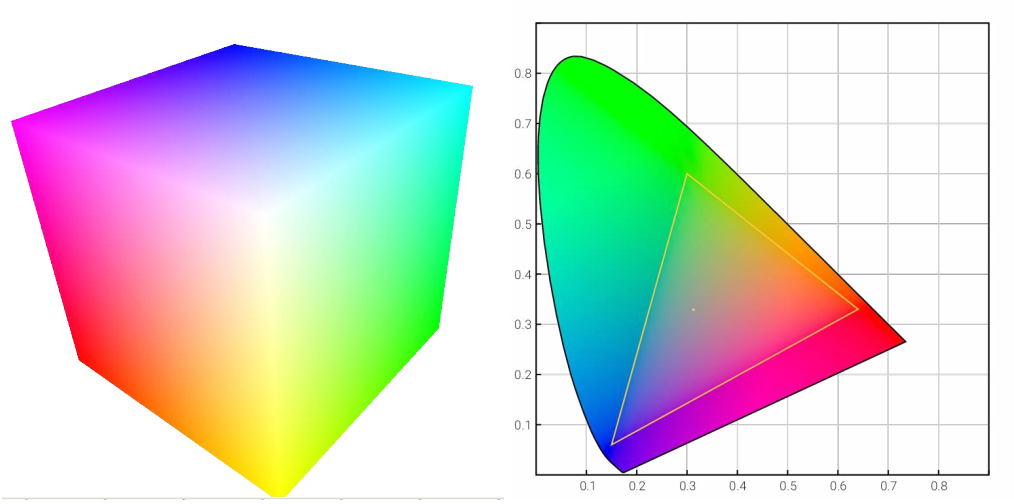


Figure 2.2: On the left Red, Green, Blue (RGB) color space cube, and on the right CIE xyz color space representations.

2.2 light modelling

Lightning model also known as Shading model or illumination model is used to calculate the intensity of light that is reflected at a given point on surface. There are three factors on which lightning effect depends on light Source, surface, and observer. **Light source** is the light emitting source, and there are three types of light sources:

- Point Sources, The source that emit rays in all directions (A bulb in a room).

- Parallel Sources Can be considered as a point source that is far from the surface, The easiest example for this type is the sun.

- Distributed Sources – Rays originate from a finite area (A tube light).

Their position, electromagnetic spectrum, and shape determine their lightning effect. While for the **surface**, when the light falls on it part of it is reflected and part of it is absorbed. Now the surface structure decides the amount of reflection and absorption of light. The position of the surface and positions of all the nearby surfaces also determine the lightning effect. While for the **observer** it's position and sensor spectrum sensitivities also affect the lightning effect. Illumination has three components Ambient, Diffuse, and Specular components. In the ambient component, we have an equal amount of light from all directions. In simple words, Ambient Illumination is the one where the source of light is indirect while The reflected intensity I_{amb} of any point on the surface is as Eq 2.1, where I_a is the ambient light intensity, and K_a is the surface ambient reflectivity and its value varies [0-1].

$$I_{amb} = K_a I_a \tag{2.1}$$

While for diffuse component, it has no dependency on the viewer, it occurs on sur-

faces that are rough or grainy. In this component, the brightness of a point depends upon the angle made by the light source and the surface. The reflected intensity I_{diff} of a point on the surface is as Eq 2.2 where the I_p is the light intensity, K_a is the surface diffuse reflectivity and its value varies [0-1], N is the surface normal, and L is the light direction.

$$I_{diff} = K_d I_p (N.L) \tag{2.2}$$

While the Specular component, it is viewer dependant, it happens when light falls on any shiny or glossy surface most of it is reflected back. The reflected intensity I_{Spec} of a point on the surface is as Eq 2.3 where the I_s is the light intensity, K_s is the surface Specular reflectivity and its value varies [0-1], and θ is the viewing angle. While $COS(\theta)$ falls off as the viewer point moves away from the surface normal.

$$I_{Spec} = I_s K_s COS(\theta) \tag{2.3}$$

Shading is referred to as the implementation of the illumination model at the pixel points or polygon surfaces of the graphics objects. The shading model is used to compute the intensities and colors to display the surface. The shading model has two primary ingredients: properties of the surface and properties of the illumination falling on it and camera position. For example, if we have a geometry model with vertices colors, we can calculate the result of light interaction with colors at vertices, therefore, use interpolation to color the interior of the faces (e.g. triangles), while different shading methods use different interpolation. Shading techniques can be divided into Flat shading, and Smooth shading (Gouraud shading, Phong shading, and others).

2.3 Rendering

Rendering is the process of generating an image from a model, by means of a software program. The model is a description of three-dimensional objects in a strictly defined language or data structure. It would contain geometry, viewpoint, texturing or coloring, and lighting information. The resulting image is referred to as the render. Rendering aims to combine this different information together to provide the image, a rendered image can be understood in terms of a number of visible features:

- Shading - how the color and brightness of a surface varies with lighting.

- Texture-mapping – a method of applying detail to surfaces.

- Bump-mapping – a method of simulating small-scale bumpiness on surfaces.

- Fogging/participating medium – how the light dims when passing through a non-clear atmosphere or air. - Shadows – the effect of obstructing light.

- Soft shadows – varying darkness caused by partially obscured light sources.

- Reflection – mirror-like or highly glossy reflection.

- Transparency (optics), transparency (graphic), or opacity – sharp transmission of light through solid objects.

- Translucency – highly scattered transmission of light through solid objects.

- Refraction – bending of light associated with transparency.

- Diffraction – bending, spreading, and interference of light passing by an object or aperture that disrupts the ray.

- Caustics (a form of indirect illumination) – reflection of light off a shiny object, or focusing of light through a transparent object, to produce bright highlights on another object.

- Depth of field – objects appear blurry or out of focus when too far in front of or behind the object in focus.

- Motion blur – objects appear blurry due to high-speed motion.

- Non-photorealistic rendering – rendering of scenes in an artistic style, intended to look like a painting or drawing.

Each of these features adding more realistic feeling to the image. One of the most important aspects of rendering is the light calculation, as known light is emitted from light source, but tracing every particle of light in a scene to render an image is almost impractical and spend an amount of time. Therefore, more efficient modeling techniques have emerged. Therefore, a few loose families of more-efficient light transport modeling techniques have emerged such as rasterization, ray casting, and ray tracing.

Rasterization, including scanline rendering, simply is geometrically projected objects in the scene to an image plane, without advanced optical effects.

While in **ray casting**, considers the scene as observed from a specific point of view, calculating the observed image based only on geometry and very basic optical laws of reflection intensity, and perhaps using Monte Carlo techniques to reduce artifacts.

And for **ray tracing** is similar to ray casting, but employs more advanced optical simulation, and usually uses Monte Carlo techniques to obtain more realistic results at a speed that is often orders of magnitude faster.

There also **radiosity** light transport technique, it is not usually implemented as a rendering technique, but instead calculates the passage of light as it leaves the light source and illuminates surfaces. Most advanced software combines two or more of the techniques to obtain good-enough results at a reasonable cost.

Chapter 3

Related Work

Scene understanding is the process of describing elements/objects in the images, the number of objects present, localization, and semantic relationships between different objects. Generally, there are two different approaches for scene understanding, the first one is the top-down approach while the other one is the bottom-up approach. Scene understanding technique is considered as a top-down approach if it is based on global information without focusing on the present objects in the scene. While the bottom-up approaches depend on low-level features like (edges, contours, corners, etc.) to find the minute details of regions (e.g. recognize objects) to understand the scene. Many computer vision algorithms are trying to extract those features (either for top-down or bottom-up) as the Computer Vision's fundamental challenge is to make computers understand things the way we humans do, moreover it is a more challenging task when it is multi-view based due to visual inconsistencies between multi-views correspondences because of camera-related issues such as different camera sensors, tuning parameters, and amount of captured light at different distances from different views in addition to specularities, and occlusions problems. In this thesis, we are taking advantage of multi-view correspondence information to overcome these Computer vision challenges and provide higher performance techniques to extract or estimate information that could serve as primary or complementary information for multi-view scene understanding methodologies. In this

Chapter, we are presenting a literature review for the Computer Vision related topics as following, Firstly, We introduce a detailed literature survey related to semantic color-name and particularly how it is perceived in the context of computer vision alongside shadow detection and removal techniques which could affect color naming in Section 3.1. Following that, in Section 3.2 a detailed literature survey related to single and multi-view specular detection and currently available datasets which could be potential for such light analysis problems. Afterward, providing accurate mesh colors and textures from reconstructed multi-view scenes related works are presented in section 3.3. Finally, a comprehensive survey on Light Source Position (LSP) detection is presented in section 3.4.

3.1 Semantic Color-Name

In this section, we are going to highlight the related works to estimating color-naming from single and multi-views alongside previous works related to shadow detection and removal as we are studying the effect of these factors on the color naming estimation process.

3.1.1 Semantic Color Naming

Early work in color naming by Berlin and Kay [31] identified eleven basic color labels shared among most human languages. One of the first work in automatic color labeling was by Lammens et al. [32] who proposed a fuzzy computational model where the membership of a set of Gaussians fitted to Berlin and Kay's labels predicted the color label. Seaborn et al. [33] fitted data from a psychophysical human experiment using fuzzy k-means to create the grouping of colors. Similarly, Mojsilovic [34] performed subjective experiments to label the colors from the

Inter-Society Color Council-National Bureau of Standards (ISCC-NBS), creating a color vocabulary based on the agreement between their labels and ISCC-NBS color names. Vandenbroek et al. [35] also used subjective human experiments, creating a similar dictionary that is then used as markers within HSI color space (Hue, Saturation, Intensity). Vandenbroek et al. then projected these markers into 2D using Fast Exact Euclidean Distance projection technique to create a complete segmentation of the color space. Benavente [36] defined the color-naming task as a decision problem using fuzzy-set theory based on the definition of triple sigmoid with an elliptical center as a membership function for the different colors. Serra et al. [37] subsequently used the work of Benavente [36] as a color descriptor over regions which could be combined with edge descriptors for image decomposition through a Markov Random Fields (MRF) to create color aware homogeneous regions. The aforementioned works in automatic color labeling are generally referred to as chip-based methods as it used color chips as training data.

Weijer et al. [38] proposed to learn from real-world images allowing more general color distributions to be learned. Their method learned a Probabilistic Latent Semantic Analysis (PLSA) topic distribution from the noisy data. Images pixels were represented as words in a LAB histogram feature and the ordered latent topics were used to provide a probability of colors. This method is still considered as the state-of-the-art in single image color classification and is commonly used to generate visual questions and answering ground truth. More recently, in contrast to other referenced methods Liu et al. [39] used the color naming as a part of fashion parsing model, they infer 13 color names and other attributes for each pixel via an MRF inference model. Similarly using MRF, Liu et al. [40] used the color-naming model proposed in [36] to build a MRF to propagate the color labels from regions under normal illumination to abnormal regions to help estimate color names.

Most recently, Cheng et al. [41] proposed an end-to-end, pixel-to-pixel Convolutional Neural Network (CNN) learned from a pedestrian color naming dataset to assign a consistent color name to regions of single object's surfaces. The color naming problem has also been considered from a multi-label perspective for ImageNet for assigning attributes to images by Russakovsky and Fei-Fei [42]. To the best of our knowledge, to the best of our knowledge, our work [43] is the only work on multi-view color labeling.

3.1.2 Shadow detection and removal

The shadow detection and removal problems were originally formulated as a physical model of illumination and color [44, 45]. These illumination invariant techniques performed well on high-quality images and with calibrated sensors but typically performed poorly on web-quality or consumer photographs. An alternative to the physical models, data-driven approaches extracted features from pixel values or regions to treat the problem as a binary classification task. Different classifiers have been used such as Support Vector Machine (SVM) [46, 47, 10], kernel Least-squares support-vector machine (LSSVM) [48, 49] and decision trees [50, 51] to label image regions into shadow and non-shadow regions. After, an MRF graph-cut with energy minimization optimization provided spatial consistency [10, 46] to the mask. Also, Conditional variants using Conditional Random Fields (CRF) [51] have been used to further improve spatial consistency.

Khan et al. [52] were the first to apply deep models for shadow detection, they used two CNN to learn deep features for shadow detection. Where one of the CNNs learned the boundary features and the other interior of the shadow. The predicted posteriors were then fed into a CRF for spatially consistent masks. End-to-end ap-

proaches were explored by Qu et al. [53] who used a three-branch network, global, appearance, and semantic, which are combined to generate a mask. Alternatively, Hu et al. [54] proposed a Direction-aware Spatial Context RNN module which models the gradients that occur at shadow boundaries to infer the shadow region within the image. Nguyen et al. [55] introduced GANs for shadow detection, proposing to use Stacked Conditional Generative Adversarial Network (scGAN) where the loss of the trained shadow detector is parameterized through adding an additional sensitivity weight provided to the generator to weight examples and avoid the issue of unbalanced training data. Wang et al. [56] used GANs in a multi-task setting considering detection and correction based on scGAN. Where the first network generates the mask and the second the shadow removed image which would be evaluated by an adversarial network. Le et al. [11] proposed GAN framework composed of two networks, a shadow attenuation network (A-Net) and a detection network (D-Net), which are jointly trained where the output of A-Net used to train D-Net and subsequently generate the shadow masks.

3.2 Specularity-highlight

There are two types for specular highlights detection Approaches: 1) based on single image and 2) based on multi-view images. This section reviews the recent literature on both approaches as well as to related datasets. We review works which explicitly or implicitly target either only to detect specular highlights or both to detect and remove them.

3.2.1 Single image specular highlights detection

The single image based approaches encompass different sub-topics related to detection, removal, reflection separation, intrinsic image decomposition, inpainting, etc. According to Artusi et al. [57] techniques can be grouped into two categories the ones related to color space analysis, and the ones to neighborhood analysis.

The techniques that use color space analysis to detect the specular highlights exploit the image colors and the chromaticity distribution in order to extract its diffuse and specular components. Klinker et al [58, 59, 60] used the RGB color space as a baseline, while in [61, 62] the authors used the normalized values of the YUV color space . On the other hand Bajcsy et al [63] proposed a new color space specially for this problem called S-space which analyze the color variations on objects in terms of brightness. But using color to detect specularities is not efficient as there are always confusion between bright colors and specularities.

Neighborhood analysis techniques appeared for the first time in the work of Weiss [64] which tried to recover the intrinsic information from image derivatives, as Weiss assumed that every image derivative is caused by either shading or reflectance since this technique relies on applying local operations on the pixel color information. The neighborhood analysis techniques were further explored in the works of Tan et al. [65, 66] and others.

Tan et al in [5] proposed to use the chromaticity of pixels to provide a specular-free image, as they proposed that the maximum the chromaticity the lower the specularity of each pixel, so they removed the specular reflection component by iteratively comparing the maximum chromaticity between neighbor pixels regardless of their colors to separate the reflection component and stop when the difference is below a

specified threshold. Shen et al [67] and Yang et al. [6] provided an enhanced version of [5] which was better in speed and accuracy, while it kept the same performance intact. In [4] the authors proposed to cluster textured surfaces pixels into clusters in the a pseudo-chromaticity space, and estimate an intensity ratio for each cluster, at the end they computed the specular reflection component of each pixel based on the intensity ratio. The work in [1] is dependent on single-color areas to separate the reflection components by calculating and separating its reflection components at the same time. Despite that they showed better results than [5], and [6], their main drawback is that they depend on manually adjusted parameters which could result in inaccuracies. Souza et al [3] proposed to cluster the pixels on the pseudo-chromaticity space, and separated the diffuse and specular components using an adaptive intensity ratio estimation.

Finally, Yamamoto et al. [7] tried to improve the separation of reflection components by using a non-linear high-emphasis filter function on each component and then improving the result by replacing erroneous pixel intensities based on the similarity with the input image. On the other hand, Lin et al [2] took advantage of the data-driven dynamic and proposed a deep model for single image specularity removal based on a GAN framework which targeted to solve most of the previous traditional techniques drawbacks. However, their model still can fail in real-life applications, while it tends to darken the images due to the simplicity of their training data.

All the above approaches are based on lab constrained scenarios, see Figure 3.1. We can notice that in all images either the background is low-brightness or totally black, while we can barely find white objects or well-lit natural images.

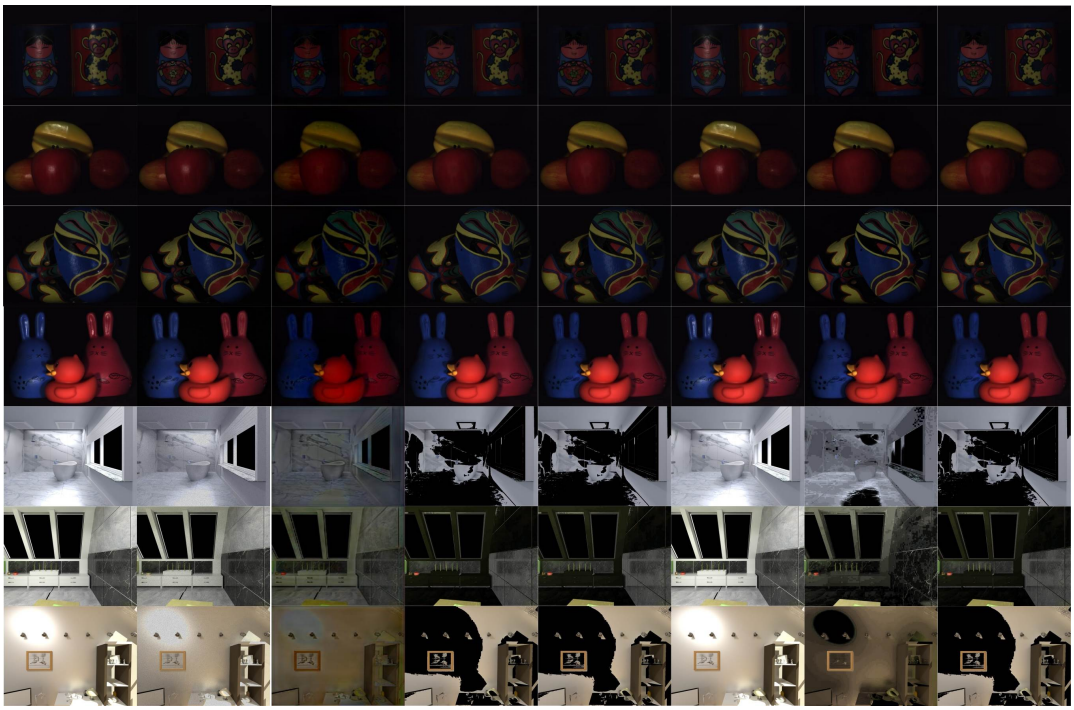


Figure 3.1: From left to right: Original image, Akashi et al [1], Lin et al [2], Souza et al[3], Shen et al [4], Tan et al [5], Yang et al [6], Yamamato et al [7]. The performance of single images specular highlight detection and removal algorithms on conventional images used in evaluation (rows 1-4) which we can refer to it as lab-constrained images, and on normal images (rows 5-7), it is clear that they are performing good on the conventional used images but they have very poor performance on natural (non lab-constrained) images.

3.2.2 Multi-view specular highlights detection

Weiss [64] proposed to remove specularities by recovering the common reflectance through computing the maximum likelihood estimate. Weiss required to have a sequence of images captured from the same scene but under different illumination conditions. Some of the first techniques used multi-view images with special lighting setup like Sang et al. [68] in which they were searching for the color differences between two images, they used binary Color Histogram Differencing (CHD) by representing both images in a binary histogram for RGB values and then calculating

the difference between the two histograms. However, this technique was prone to occlusion as it did not consider geometrical information, which finally could cause false detection of reflections.

In Lin et al. [12] they overcame the occlusion problem by using consensus from multiple sets of three images (left-center-right) (called Multi-CHD) instead of using two images at a time. Chen et al. [69] required nearly 200 images of a surface from different views to be able to reconstruct the specular component which makes it unpractical. Other techniques [70, 71, 72, 73, 74] introduced solutions based on different polarization orientations in multi-view images. Images were taken with a multi-flash setup from the same point of view introduced by Feris et al. [75], while Agrawal et al. [76] used a couple of flash/no-flash images to separate reflections. [77, 78] required a specific scene illumination setup (distant environmental illumination), their work relies on a specific high illumination frequency in order to apply high-low frequency separation.

Finally, in more recent works Kobayashi et al. [79] and Takechi et al. [80] proposed to have images of an object captured from a fixed viewpoint using multi-spectral and multi-directional light sources with a known color and direction or to make use of light fields respectively.

Multi-view specular detection is also examined in [81, 82, 83, 84, 85, 86, 87, 88] where highlight detection and/or removal is applied in sequenced images (i.e. videos) which is not the case which we target here in our thesis.

3.2.3 Potential light analysis Datasets

There too many datasets used in the Computer Vision field, in this section we will discuss their validity for light analysis. We will try to mention the most related

datasets which could serve light analysis problems. There are several proposed annotated 3D datasets targeted to study several computer vision problems like objects localization, segmentation, recognition, and others which have information such as RGB-images, depth, normals, instance object annotation, and semantic object class annotations which can serve for such tasks, but for light analysis purpose the available datasets are not valid enough to serve for this purpose, e.g. in Interiornet dataset [89], the authors provided albedo, depth, normals, illumination, fisheye, panorama, etc. They mentioned in their work that light source information and the interactive simulator (ViSim) program will be available, but unfortunately, none of these two entities is available nor reachable. Moreover, it seems that there are problems in rendered images which are missing specularities while the generated illumination maps have colors while it should not be (see Figure 3.2). All of these drawbacks and unreachable information making this dataset not valid for light analysis.

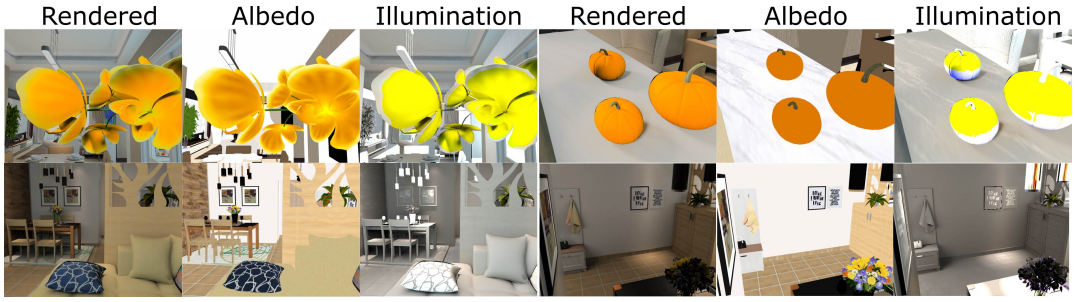


Figure 3.2: Wrong illumination map examples from Interiornet dataset.

McCormac et al[90] proposed the SceneNet3D dataset which has a wide variation in objects, objects position, and light-sources positions which make it a good potential dataset for light analysis. Despite that Depth, normals, class, and instance object annotations information are provided, but unfortunately, the light information like light source position, type, or intensity information is not provided. Mostly we can say that all of the used light sources in it are point-type light sources. Honauer et al

[91] proposed the 4D Light Field Benchmark which is valid for studying light-fields. There are also some works proposed some limited individual images as evaluation datasets (varying from two to seven) are provided in [5, 92, 4]. These include both ground truth specular-free or not (useful only for the reflection detection use case) images, see Figure 3.1, rows 1-4. However, as mentioned earlier these images are heavily lab constrained and not suitable for evaluating real life scenarios. There are also similar limited works targeted other individual light problems like a shadow in Guo et al. [10], Vicente et al. [93]. Others like [94, 95] who proposed datasets with HDR images only to estimate scene lighting without any further annotations regard to materials, the geometry of the scene. Murmann et al [96] proposed a dataset of single images with multi-illumination and different light conditions, but unfortunately, they only provided HDR images and material type annotation (14-type), but neither reflection components annotation, nor geometry information is available. Lin et al. [2] generated a synthetic dataset for specular highlight detection and removal, but it is not publicly available while it is again limited to images containing only eight objects (three for training and five for testing), while each image contains only single object located in its center, and all light sources are directed to that object. While they mentioned that their model has drawbacks and can fail in real-life applications, they supposed that this could be because of the simplicity of their data, especially the lack of a background.

On the other hand, our proposed LIGHTS dataset provides many scenes (18) where each includes many objects with physically-based rendered materials. Moreover we provide wide numbers of details and different lighting setups refer to section 5.1 for details. By default each object in the scene is part of a larger environment with surroundings, meaning that we have a wide variation in the background. We created it to be suitable especially for light analysis purposes. We used path tracing light

transportation in order to simulate real-world light behavior and render multi-view images with reflections as realistic as possible.

3.3 3D Mesh-Colors

Providing accurate mesh textures for reconstructed multi-view scenes has been tackled in many works as per-vertex texturing (one color per vertex) or by providing texture images for faces. It was clear from the early methods that there is a problem in providing consistent colors for the mesh from multi-view images as averaging colors from all views like Frahm et al. [97] is providing inconsistent colors to texture the mesh which in turn cause visual inconsistent colors as well as seams to be visible between textures parts. This problem has been mentioned implicitly under different names to correct the color inconsistency problem from multi-view images like mesh texturing, solving seams between textures, visual consistency colors, or seamless color mapping. In [98, 99] they tried to solve the problem of consistent colors in their textures by estimating per-vertex texturing (one color per vertex), but they have undergone ghosting and blurring artifacts due to the misaligned errors, moreover, in [99] they are depending on special scene assumption, in which scene should be captured under cloudy and not cloudy environment situations to solve the problem. Later, Zhou et al.[100] proposed an expensive color map optimization algorithm for 3D reconstruction as a per-vertex texturing algorithm which could work for small reconstructed objects, but it is not practical for large scenes. This problem also tackled in [101, 13, 102] as adjustment of color discontinuities to hide seams, especially in Waechter et al. [13], they proposed impressive work to solve this problem by modifying on Lempitsky et al. [103] technique, but still, they could not provide high-resolution results as they were undergone to texturing problems and some in-

consistencies in registering textures to mesh. Despite that the most recent techniques for mesh texturing like [104, 14] are providing better textures, but they are still suffering from inconsistent color estimation from multi-view problem as they depend on the best-view selection which is prone single views problems, while in our work we estimate the color through aggregation process from all frames. Also in [105], they selected the best-view to provide high detailed texture, but later they smoothed between textures to hide seams, but not fix the main problem which is inconsistent colors for textures that caused these seams. In this thesis, we are separating the process of consistent color estimation from the mesh texturing process. We also showed that the estimated consistent colors could be used to correct already textured model colors. To be clear we have to differentiate between works that tend to create 3D painting tools for artists to work with like Ptex [106] and Mesh colors [107] and image-based modeling in vision applications which depend on texture maps like mentioned techniques in related work.

3.4 Light Source Position

Regard the light source estimation problem, Dejan et al. [8] estimated the illumination and material in indoor scenes by using an invertible light transport simulation. They proposed an optimization method for optimizing for physically correct light transport and material models using Stochastic Gradient Descent (SGD), but this approach fails if the light source position falls in a missing mesh part of the scene. From the speed point of view, this approach is mainly affected by the number of faces. From the estimated albedo which seems to be not well estimated, we can indicate that the technique is not working properly, see Figure 3.3.

Figure 3.3: From Dejan et al [8] paper, From left to right Input photographs, rendering, Albedo. It is clear in the estimated albedo that the regions in the wall under shadow has wrong estimated albedo which is different from the wall, which mean that the technique is not solving the problem as expected.

Moreover, it is not mentioned how to compensate for the missing information about tone-mapping calibration in cameras which is important in their case to estimate material reflectance properties. Li et al. [108] proposed an Inverse Rendering framework based on a deep model to estimate individual scene components like spatially-varying lighting, spatially varying non-Lambertian surface reflectance, and shape. Gardner et al. [95] proposed to use an end-to-end deep model to regress a Low Dynamic Range (LDR) image to High Dynamic Range (HDR) illumination to infer light source positions. Other deep models have been proposed to estimate related parameters [109, 110] which try to estimate material and environment map which could be useful, but they tested only on single objects. While in Gardner et al.[95], they trained deep model to estimate scene illumination and the location of lights in a scene from a single limited field-of-view photo. The main noticeable drawback in deep models or trained classifiers like [111] in dealing with light problem like light source position estimation or specularity detection or similar problems, that they

tend to classify the white object under high or even good illumination environment as light source, this is because of the light behavior which do not have a geometrical shape or unique effect or color in all circumstances to be learned properly, based on this information, the solution for such a problem should be through physical-based solution. Boom et al. [112] proposed a graphical processing unit based method to estimate a point light source position in a scene recorded by an RGB-D camera, but they reported that technique with an average error of 20 degrees. Zhang et al. [113] detected light source position based on shadow clue, which is not guaranteed to be clear and available every time. Edward et al. [114] proposed a system to identify the locations of light emitters in the scene in a semi-automated fashion based on using predefined architectural features including walls, ceiling, and others. Finally, they based on methods similar to radiosity based formulations of inverse lighting to recover lighting and diffuse reflectance parameters. Jorge et al. [115] tried to estimate the positions, directions, and relative intensities of the light source from a single image. They used an object within the image as a probe, by removing the highlights and noise from its borderline, then dividing it into segments and estimating their albedo factor to later add them to line segments to remove its texture. They infer the number of light sources and to detect their position in screen space, and finally in 3D space or at least its direction, but they required an interaction from the user to choose an arbitrary object to use as a probe. Moreover, they assumed that the object is convex. Other works like [116, 117, 118] used known geometry information and relied on it to locate critical points, which are points at the boundary of surface areas that are affected by a different combination of lights. However, these techniques require using Lambertian sphere as a calibration object or detecting directional lights only.

Chapter 4

Multi-View Color Naming

This chapter explores the classification the color attribute of objects using multiple images of the same objects. This problem is more complicated than the single image estimation because varying environmental effects, such as, specularities or shadows from light sources, can result in deficient accuracy. These effects depend principally on the camera positions and the type of objects material. Single image techniques concentrate on improving the inequity of between colors, but in multi-view systems additional geometric knowledge is available, but it should be used wisely. To this end, we propose three methods to aggregate image pixel information in multi-view that increase the performance of color name classification. Moreover, we analyze the effect of shadows by employing automatic shadow correction and detection techniques on the color naming problem. We examined our proposed methods on a new multi-view color names dataset called Matterport 3D Color Naming (M3DCN) which contains outdoor and indoor objects. Furthermore, we experimentally demonstrate that addressing visual outliers in multi-view images such as shadows enhances the performance of the color attribute in the decision process.

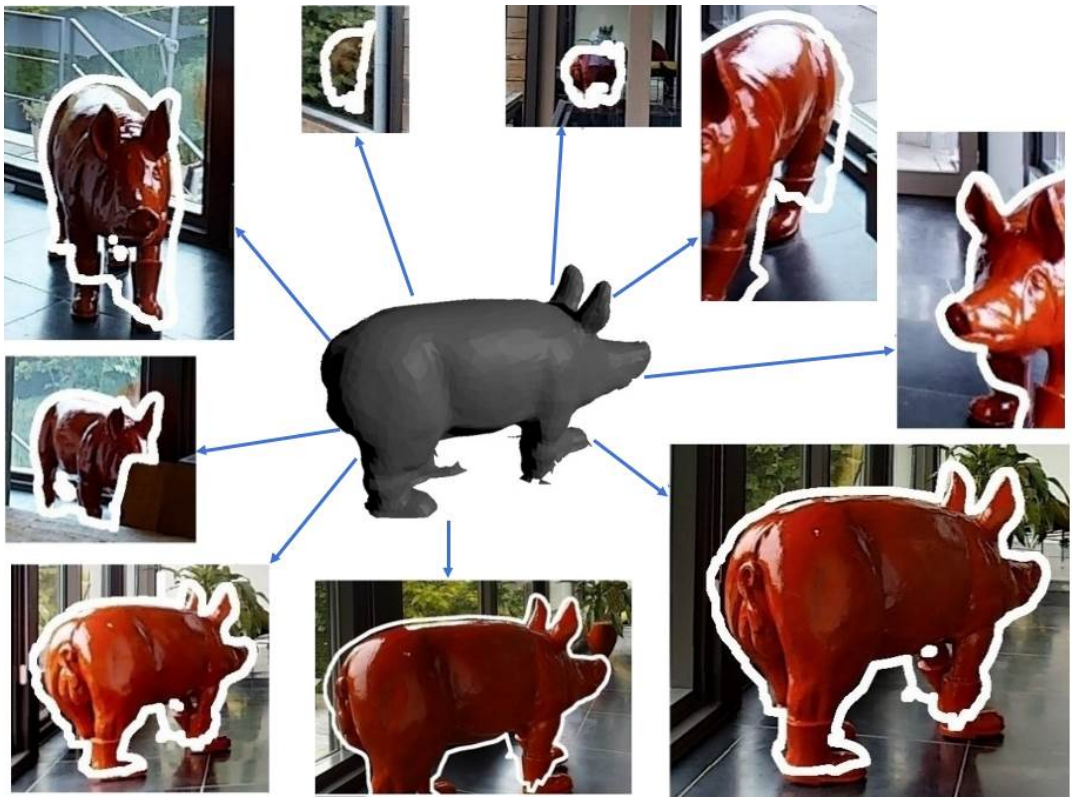


Figure 4.1: 3D mesh of an object from Matterport 3D dataset [9] in center, with a subset of views surrounding. The surrounding views demonstrate different scales, specular reflections, and self shadowing.

The classification of color is used for applications such as pedestrian re-identification, visual question and answering, automatic visual captioning, and many other computer vision tasks [41, 119] wherever the interpretation of the semantic attribute "color" is fundamental. For example, in content-based retrieval, deciding the color of the object in an image restricts the possible search space. Contrarily from the single view issue, color naming of an object ends in a more complex task. 3D reconstruction [120] is able to help to inform the decision of color, while integration is nontrivial. Various views lead to changes in illumination given both by the position of the light sources in the scene and material of the objects. Moreover, when seeing an object (Figure 4.1) with a wide camera model, occlusions might provide a limited

view of the object or (self-) shadows could hide the color.

Thus estimating the correct color in case of having a set of possibly noisy observations is a challenging open problem and new. Previous research tried to solve the problem in single-view by proposing discriminative color algorithms [38, 36, 33, 32, 41], but outcomes are still unsatisfactory on in-the-wild datasets. This is due to two main reasons.

First, illumination colors are always influenced and affected by the environment lighting, and an object can be observed as having distinct colors based on the nearby or surrounding light and its material reflection properties. As an example, in low-illumination or shadows, low degree or dark-colors can be observed by humans as black-colors, while white-colors would be observed as gray-colors while other colors will tend to be less brighter or darker. However, if we have previous knowledge that a particular object is affected by shadow, this problem would be simpler. For this reason, color naming may possibly be largely facilitated by getting the regions where the object is affected by shadow. In addition, if the object affected by specular highlights given its reflectivity, the corresponding regions will also cause difficulties in deciding the color names.

Further, colors ambiguity boundaries among colors in color space are not properly quantified e.g. humans can not confidently decide what is the color name of the color in the space among green and blue. As the problem is unclear for humans, this difficulty and ambigious is translated to any computational approach.

In this chapter we are trying to resolve the multi-view color naming problem by offering aggregation approaches to exploit the data provided by the multiple images. In particular, we define three strategies for color information merging while attempting to reduce outlying effects (e.g. shadows) and noise. To this end, before starting

the color labeling process, we attempt to obtain prior information about shadows regions, in order to either correct the shadow pixels or to remove them before our multi-view decision process.

We can list the main contributions in this chapter as:

- We offer for the first time three simple and robust aggregation methods for multi-view color naming that exploits the available information across views;
- We use single image shadow detection and correction techniques to analyze the effect of shadows on multi-view color labeling and propose to use shadow information in the aggregation techniques to enhance the performance;
- Present Matterport3D Color Naming dataset (M3DCN), the first multi-view 3D dataset for color naming[1].

[1] M3DCN available at https://github.com/mohamed-elkhouly/M3DCN

4.1 Multi-View Color Naming Aggregation

We define the color-labeling problem in multi-view data as a data aggregation challenge, in which the auxiliary information from multi-views can alleviate specularities and shadows to achieve improved accuracy in classification. We, therefore, base our work on using PLSA from Weijer et al. [38] which is the state-of-the-art in single image color classification, and extend it to employ different approaches to aggregate the information among different views. At first, we get our multi-view data in two forms dense and sparse points (described in subsec. 4.1.2) from the M3DCN (subsec. 4.2). Then, the classification is performed over 11 colors, to match the prior works, we outline the method of Weijer et al. [38] (subsec. 4.1.1). We then define three methods for aggregating this information across multi-views (subsec. 4.1.2), finally we address a common environmental issue of shadows that could cause significant changes in the classification decision. The full system pipeline shown in (Figure 4.2).

4.1.1 Color Naming

Color classification is often handled as a topic learning problem, as in [38] which they learn from weakly labeled images employing a variant of the Probabilistic Latent Semantic Analysis (PLSA) model. Given a set of images $I = \{i_0, ..., i_N\}$ in $L * A * B$ color space, the set of pixels for I_j form a frequency histogram by quantizing the color space by $[10, 20, 20]$ of $L * A * B$ channels respectively. Where each histogram represents a document $D = \{d_0, ..., d_N\}$, as in traditional text analysis notation. Therefore, the set of words $W = w_0, ..., w_m$ refer to the bins of the color space quantization. Then, PLSA optimizes a set of latent topics $Z = \{z_0, ..., z_J\}$ through expectation maximization of the conditional probabilities as in the equation

(Eq. 4.1):

$$p(w|d) = \sum_{z}^{z \in Z} p(w|z)p(z|d), \tag{4.1}$$

where $p(z|d)$ and $p(w|z)$ are multi-nomial distributions (notation as per [38]). Weijer et al. proposed two ways to exploit the learned topics, using an indexed look-up of the word color probabilities (Eq. 4.2) and utilizing a region prior. Given our multi-view scenario, we take advantage of the object segmentation, and therefore the prior is not needed. The indexed approach is referred to by PLSA-ind:

$$p(z|w) \propto p(z)p(w|z). \tag{4.2}$$

Where the prior over the color names $p(z)$ is taken to be uniform.

4.1.2 Multi-View Color Naming aggregation

Given a set of images I, we propose three methods for aggregating the multi-views of a given object. The object views $V = \{v_1, ..., v_r\}$ where $V \subseteq I$ provides the color image, object 3D point cloud , and the object mask. We propose to use sparse points that are projected from the 3D point-cloud into the color image; and dense points which acquired by filtering the color image with the object masks. The corresponding $L * A * B$ values of the 3D points and their related word then form a decisions C_p we use to refer to the distribution $p(w|z)$, and C_v to refer to the color decision for frame. The three methods are as follows:

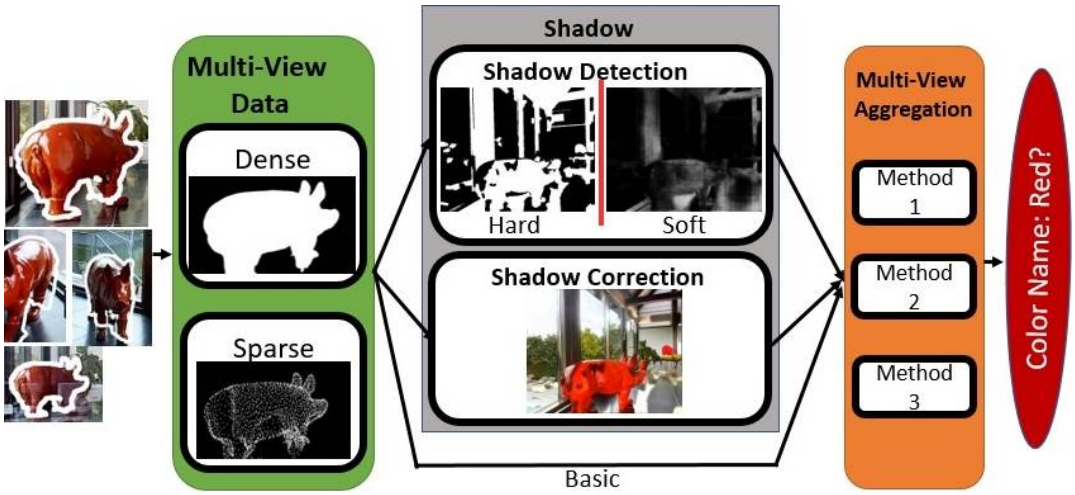


Figure 4.2: Multi-view aggregation for color naming pipeline using the object multi-view data. Both dense and sparse points representation of the object are considered in the performed experiments. Then the three different aggregation methods are experimented. Firstly, data is passed to test on the aggregation techniques directly (basic). Secondly, shadow detection by using either soft or hard masks is applied, we exclude the data pixels labeled as shadow. The third one, is to apply a shadow correction technique, then use the corrected data to be test on the aggregation techniques. After All, the output is the color name of the object.

Method 1: Follows PLSA-ind [38] for predicting the color name of the object in each view v_i, later aggregated by the most frequent across views:

$$C_v = mode\{argmax_z(C_{pk}) : k = 1, \ldots, n\}, \tag{4.3}$$

$$C_{Object} = mode\{C_{vl} : l = 1, \ldots, r\}, \tag{4.4}$$

where n is the count of object pixels in the given frame, C_{Object} is the color name decision from all frames.

Method 2: Uses all probabilities and therefore not applying max which limits the

propagation of information. Defined as:

$$C_v = argmax_z(\sum_{k=1}^{n}\{C_{pk}\}), \tag{4.5}$$

$$C_{Object} = mode\{C_{vl} : l = 1, \ldots, r\}, \tag{4.6}$$

The difference between Method 1 and 2 can be seen in Figure. 4.3, which shows how intra-class confusion can be lost at this initial stage.

Method 3: As with Method 2 the probability distribution C_p is used, in addition to taking into consideration an importance for each frame in the decision process. This importance can be calculated by the observable area of the object at each frame, using it as a weighting function. It is defined as:

$$C_{Object} = argmax_z(\sum_{l=1}^{r}\omega_l \sum_{k=1}^{n} C_{pk}), \tag{4.7}$$

where ω is a l_1 normalized vector comprises the weight for each frame according to the observable area of the object.

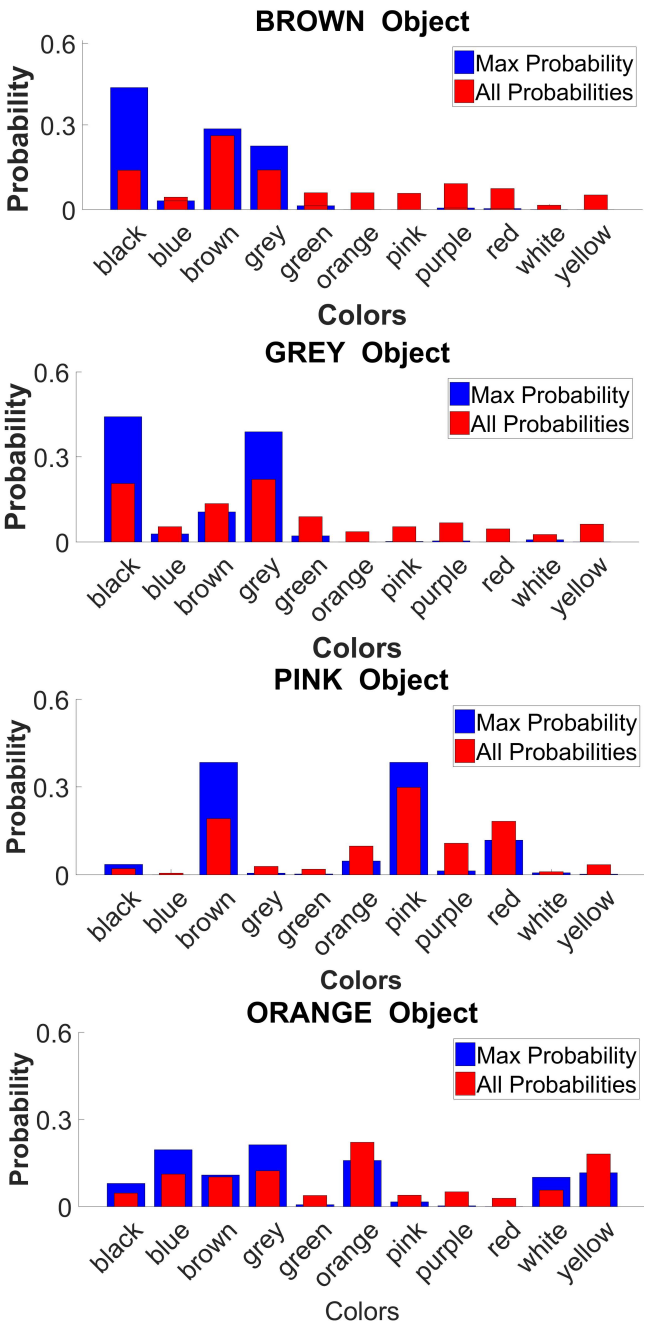


Figure 4.3: The difference between using probabilities of color names of four objects from all of their frames compared to using the maximum probability at each color name index for deciding the color name of the object. Using all probabilities (in red) is providing more information about all colors, and is better for the most occurring colors in all views even if it is not the maximum, which leads to the right label.

4.1.3 Using Shadow Detection and Removal in Color Naming

For avoiding including outliers due to shadows on the object in classification we include shadow detection. We use three types of shadow analysis, hard shadow mask a binary mask of areas in shadow; soft shadow mask a probability distribution over the image; and shadow correction for removing the effect of shadow on color within the shadowed regions. We depend on the output of Le et al. [11], and Guo et al. [10] for shadow detection, and Guo et al. [10] for shadow correction.

Soft Shadow Mask: The posterior probabilities resulted from the shadow detection algorithm defining how much each pixel is affected by shadow. The GAN-based framework ADNet of Le et al. [11] generates soft shadow masks could be used as a shadow weighting for color name classification C_{pk}. Their technique composed of a shadow detection network (D-net) and attenuation network (A-Net). The A-Net mainly was used to generate training samples for augmenting the training data for the other network (D-Net) with hard-to-predict cases to fool it. At the end, generating a soft classification shadow mask. For using the soft shadow mask as a weighting we had to normalize the soft mask and use it as a probability of shadow.

Hard Shadow Mask: It is a binary mask labels, that defines each pixel as either affected or not by shadow. Guo et al. [10] presented a Graph Cut based method that provide a binary segmentation, as the resultant optimization either falls into the sink or source. In Guo et al. approach they are classifying individual regions subsequently pairing regions to construct an MRF graph. The unary term is resulting by classifying the paired regions to *different illumination* and *same illumination* according to their likeness on appearance and textures while the pairwise term is

considered as the distance within the image between regions. The energy minimization is applied to solve for binary labeling. Besides, we can apply thresholds to the soft shadow proposal of Le et al. [11] to obtain the hard masks, using three thresholds $[0.9, 0.6, 0.3]$ to neglect shadows under the threshold (see Figure. 4.4). Other thresholds were empirically experimented and had similar results, for simplicity we only mention three levels.

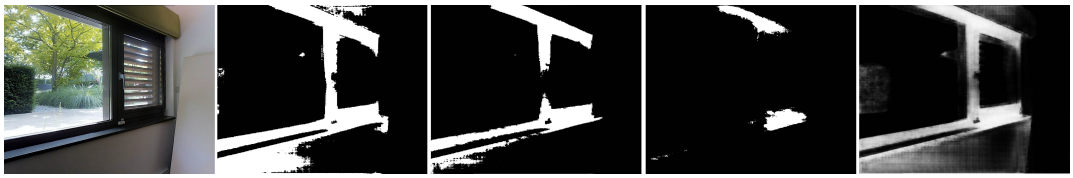


Figure 4.4: From left to right: the original image, hard shadow mask obtained from soft shadow mask using the thresholds 0.3, 0.6, and 0.9 respectively. The most right is the obtained soft shadow mask, where shadow is represented as white label pixels

Shadow rejection condition: As dark objects are often misclassified as in shadow (see Figure. 4.6), also noted in [54]. Therefore, we apply a condition for deciding when to employ shadow masks, where if $> 70\%$ of the object is identified as in shadow, the weighting will not be applied for that image.

Shadow Correction: For correcting the color of regions under the shadows mask we employ the technique of [10]. For a given region pair R_i (as described in Hard Shadow part) which falls under two sources of lights, environmental L_e and direct L_d, see Eq. (4.8). They correct the shadow regions pairs which identified as *same illumination* using the higher illumination with the assumption the illumination conditions are preferred. Constrained by the ratio of direct to environmental light in each color channel and estimating a fractional shadow coefficient value using a matting technique, enabled them to recover the shadow-free regions, see [10] for more

details. In our technique, we study the use of shadow free images $I_i^{shadowfree}$ (or more accurately shadow corrected images) instead of original captured images on the color naming process.

$$I_i^{shadowfree} = (L_d cos\theta_i + L_e)R_i, \tag{4.8}$$

As the shadow correction technique relies on the same method as in hard shadow masks [10], we employ the same shadow rejection condition to improve the correction results.

4.2 Matterport3D Color Naming Dataset (M3DCN)

We propose the Matterport 3D Color-Naming dataset (M3DCN) for multi-view color-naming which is based on objects from the Matterport 3D dataset [9]. M3DCN was collected by requesting from 7 participants to annotate the object's color-names from the Matterport 3D dataset. We presented to each participant the visible parts of objects from all views which were outlined using the object mask to make it obvious which object to focus on (see Figure. 4.1). Then, we use the objects consensus with uniform color-label in our proposed dataset which was class balanced resulting in ten (10) objects per color, (110) total. Despite that some categories are dominant, but *orange* and *pink* objects are seldom present in the Matterport3D scenes. M3DCN objects vary in number of views and material (see Figure. 4.5). We show the semantic class distribution of M3DCN objects in Table 4.1. For conciseness in the table, we had to group similar objects under one label e.g. barbers_chair or sofa_chair are grouped under the chair.

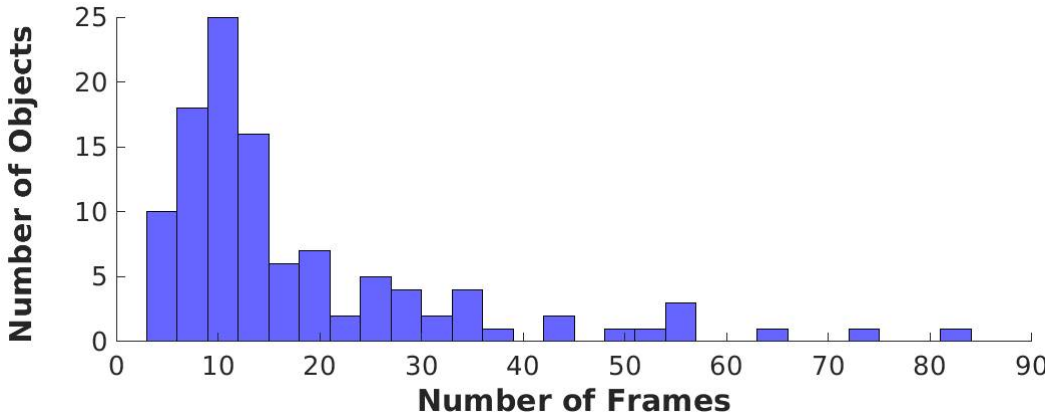


Figure 4.5: Statistics for the frequency of images (V) for the 110 objects in the M3DCN dataset.

Table 4.1: Frequency of semantic classes in M3DCN dataset.

Object	**Count**	**Object**	**Count**	**Object**	**Count**	**Object**	**Count**
fireplace	1	canister	1	picture	2	toilet	1
bathtub	1	floor	3	lamp	3	sink	1
trashcan	3	door	2	grass	1	stool	1
bed_sheet	1	vanity	1	guitar	1	kettle	1
door_frame	1	table	3	tv	1	pan	1
decoration	4	chair	17	bed	2	pot	2
clothes	1	pool	1	curtain	1	toaster	1
cushion	3	pillow	13	stair	1	plant	3
container	1	recliner	2	towel	1	cabinet	7
light	1	chaise	1	stand	1	wall	9
concrete	1	kitchen island	1	couch	6	**Total**	**110**

4.3 Experimental Results and Analysis

We extensively evaluate the various permutations of the three proposed aggregation methods and the impact of incorporating soft and hard shadows for both dense and sparse points as shown in Table 4.2 by classification accuracy of these various configurations.

When considering the use of dense and sparse points as input to the multi-view

classification within our basic approach (no shadow detection nor correction), it is clear in two of the three cases there is an improvement by using dense points with substantial improvement for Method 1 and Method 3 with 5% and 4% respectively. However Method 2 suffers a decrease of 2% caused by the sensitivity to the size of the object and an increase in intensity of noise present by considering all posterior probabilities. This behavior is generally observed across other experiments considering shadow correction or detection.

When applying shadow detection we display results with and without shadow rejection condition (latter one is shown in parentheses). When applying shadow detection, it improves the performance for sparse on average for method 2 and method 3 in sparse with 0.4% and 1.5% increasing on average respectively across all shadow detection methods and thresholds. In general, it can be seen that it either enhances the performance or has consistent performance. In contrast for using dense 3D points, there is a consistent improvement with 0.5%, 0.9% and 1.5% across our three methods with almost all methods and configurations improving on the basic performance.

While using hard shadows from Guo et al. [10] or our threshold variants of Le et al. [11], in the majority of cases there is an enhancement in contrast to the basic methods. Guo et al. can be observed to consistently improve the accuracy, alternatively Le et al. is vulnerable to the threshold being applied. Soft shadows enhance on the basic method for both types of points (sparse, and dense) but can be enhanced by a carefully selected threshold. Sample confusion matrices for weighting and omitting points based on hard and soft shadow detection respectively can be seen in Figure. 4.7, we evaluate using hard shadow masks against soft shadow masks for Method 3. In the confusion matrices, it can be noticed that colors of objects that are generally lighter (grey, orange, white and pink) are misclassified in the case of soft

shadow with their darker equivelance, e.g., white to gray. This can be clarified by the explicit hard masks being more aggressive and being spatially consistent by the MRF. It is remarkable to note that, while applying the shadow rejection condition (without parentheses), in most cases it demonstrates significant enhancement except in the cases highlighted in blue, which are using shadow correction. In these cases, the over-saturated colors from the correction makes classification easier even if it does not create visually appealing results.

Overall Method 2 is constantly better than Method 1 except in two cases (in red), while Method 3 is constantly better than both Method 1 and 2 in all cases. Demonstrating that using per point probabilities (not max or mode) and the observable area weighting of objects helps to come to a trustworthy classification. Method 3 with shadows provides enhancement of 11.8% for dense and 10% for sparse over the lowest performing basic methods demonstrating a benefit using multiple views.

Figure 4.6: A “black floor” and its produced shadow masks, from left to right: original image, hard shadow mask by using [10], and soft shadow mask by using [11]. As illustrated it is clear that it is labeled as in shadow. It is also obvious that [11] less influenced by this problem, this is obvious in the dark grey wall that appear on the middle of the right half.

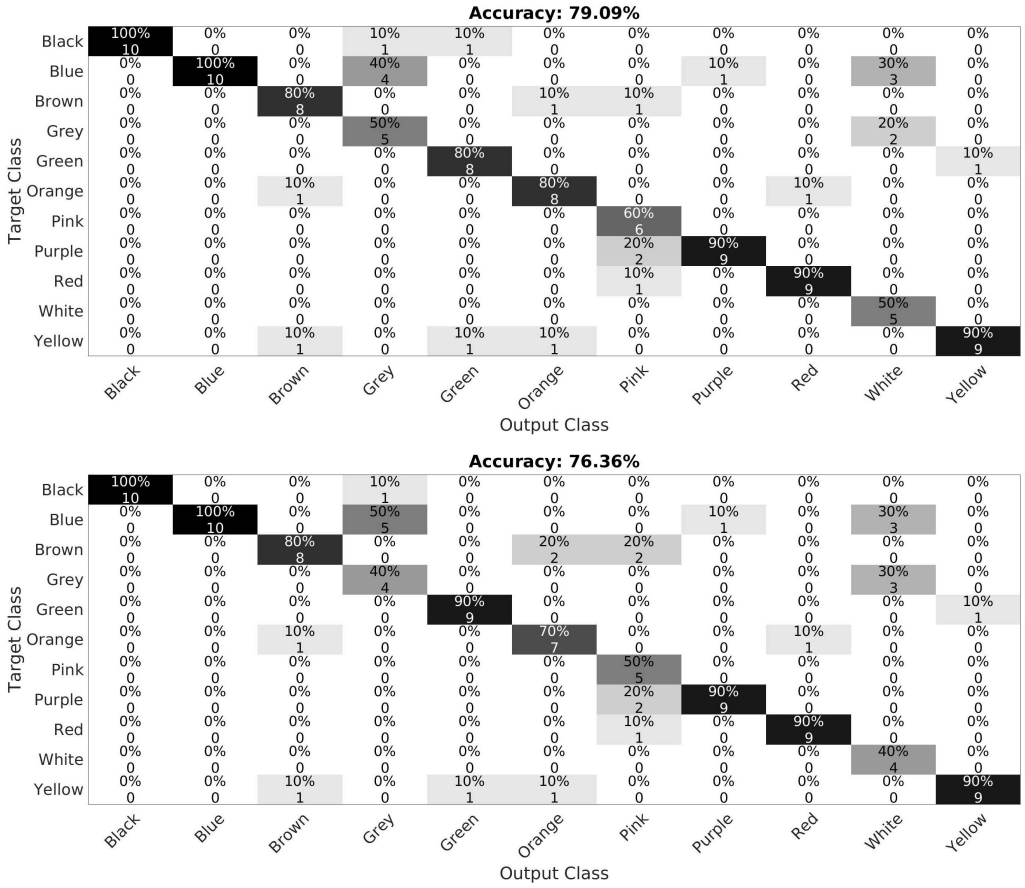


Figure 4.7: Confusion matrix for applying hard shadow masks (Up) and soft shadow masks (Down) for shadow detection utilizing dense points on Method 3.

Table 4.2: Classification accuracy of the presented three methods of aggregation using shadow detection (hard/soft), shadow correction and sparse and dense 3D object points. Highlighted in red and blue are important results analyzed in Sec 4.3 and in parentheses are experiments without the shadow rejection condition.

Testset				M3DCN		
Aggregation Method				Method 1	Method 2	Method 3
Sparse	Basic			63.64	69.09	70.91
	Using Shadow Detection	hard shadow	Guo	66.36 (56.36)	70.91 (60.91)	**73.64** (66.36)
			ADNet Threshold=0.3	66.36 (63.64)	70.00 (70.00)	70.91 (70.00)
			ADNet Threshold=0.6	72.73 (64.55)	68.18 (68.18)	72.73 (71.82)
			ADNet Threshold=0.9	63.64 (63.64)	69.09 (69.09)	70.91 (70.91)
		soft shadow	ADNet	63.64	69.09	**73.64**
	Using Shadow Correction		Guo	67.27 (70.91)	68.18 (72.73)	71.82 (74.55)
Dense	Basic			67.27	67.27	75.45
	Using Shadow Detection	hard shadow	Guo	68.18 (59.09)	69.09 (64.55)	**79.09** (67.27)
			ADNet Threshold=0.3	68.18 (67.27)	69.09 (70.00)	76.36 (71.82)
			ADNet Threshold=0.6	67.27 (67.27)	67.27 (65.45)	76.36 (67.27)
			ADNet Threshold=0.9	68.18 (68.18)	67.27 (67.27)	76.36 (76.36)
		soft shadow	ADNet	67.27	68.18	76.36
	Using shadow Correction		Guo	68.18 (70.91)	68.18 (71.82)	78.18 (77.27)

4.4 Conclusions

We propose three methods to solve the multi-view aggregating of decisions for the color naming problem. We have shown how the outliers caused by shadows affect the ability to classify color, and how excluding or correcting the affected pixels improves the classification accuracy. Testing on our proposed dataset M3DCN dataset, we achieved an accuracy of 79.09%. The proposed methods can be further extended to consider other outliers like specular highlights. While the methods are ammeanable to be applied to other probabilistic visual attribute problem in multiple views.

Chapter 5

Detecting Specularities Multi-View

In this chapter, we are detecting specularities in multi-view scenes. Depending on the task specular reflections can mislead the image interpretation. Recently, state of the art single image methods have demonstrated remarkably high performance at detecting and removing specular reflections from colored images. These methods are evaluated on limited number of images which have specific setup. Therefore, their performance are not evaluated for natural images which are captured in the wild. In this chapter, we presents a novel physically based rendering LIGHT Specularity Dataset (LIGHTS), and a simple and fast face-based multi-view specularity detection technique. It is based on exploiting single image specular highlight detectors to get potential diffuse and specular highlights in single images. We then aggregate the information from multi-views to detect the corresponding specular faces in the 3D space. Experimental results demonstrate the effectiveness of our proposed method by 3.5% improved accuracy over other multi-view technique.

Specular highlight reflections is the bright area of light that appears on shiny objects when illuminated. It can be categorized as a) direct based, in which the source of specularities is the light source itself or b) indirect based, in which light is reflected from another surface. Moreover, it can provide us with information about light source like shape, color, number of light sources, and direction (see Figure 5.1). Specular highlight reflections can only be seen if the viewer angle θ is equal

Figure 5.1: Specular highlight examples on different objects. We can see clearly from the examples how many visible light sources reflections, there are two, two, and three light sources visible on the images from left to right respectively. It is also clear in the right most image that the shape of the light sources is rectangular, while two of them are white-colored and the other one is yellow. At the same image we can also see an example of indirect specularity example, in which the ball is reflecting the wood surface under it.

to the incident light ray angle α while diffuse reflections can be seen from all other directions (see Figure 5.2). This is an important feature for detecting specularities in multi-view scenes as specular reflection brightness value for a 3D point is higher than its diffuse reflection value in other views as shown in Figure 5.3.

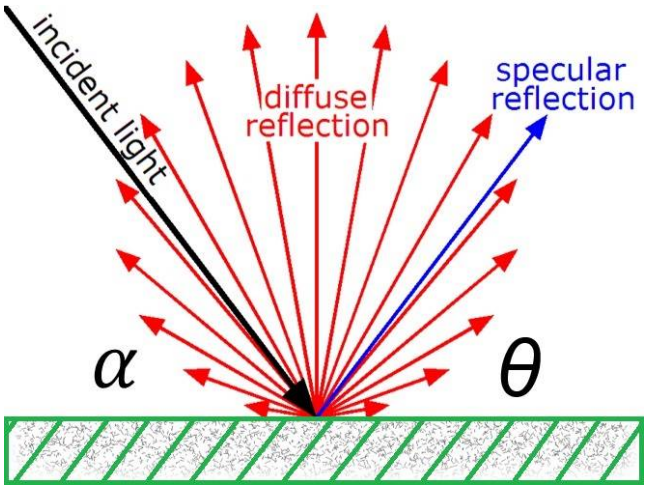


Figure 5.2: Surface example (horizontal) and incident and reflected light rays. While θ and α are the angles between the surface and the viewer ray and the incident light ray respectively.

Based on different applications, specular reflections can be useful for a specific task

Figure 5.3: Three images of the same object from different view-points. The object part inside the green rectangle of the middle image is specular while other rectangles in other images contain diffuse reflections, it is obvious that the middle one has higher brightness level than diffuse ones

e.g. light source modeling such as [121] in which they were able to reconstruct the environment from hand-held RGBD sensor and they were able to reconstruct surface light field. However, in general specularities cause problems by disrupting the results, e.g. specular highlights on the objects surface are typically treated as outliers when modeling the color properties and mostly ignored when modeling the surface opting to apply a lambertian assumption to the surface. Such an assumption leads to inaccuracies when estimating the surface properties, such as color, as depending on the material the degree of the highlight might be vast and in turn drastically affects the inferred properties. This problem is particularly evident when a single observation of the surface is used to infer its properties, however, from multiple views the specular highlight can be more easily addressed.

One reason to avoid specular highlights is the difficulty in creating a real world dataset with accurate annotation ground-truth for this phenomena. In real-world images the multitude of light sources within a scene make accurate human annotation challenging, while existing 3D datasets such as Matterport3D [122] or ScanNet [123] would be considered too uncontrolled in terms of lighting and capture setup to create an accurate groundtruth according to unknown important information e.g. light sources, and its illumination. This leads to the point where specular

detection techniques are focusing on lab setting captured for single images (shown in Figure. 5.4) or bespoke capture in the case of multi-view [64, 124, 125, 83], making the latter unsuitable for general purpose applications. Alternatively, synthetic datasets such as SunCG [126] or InteriorNet [89] lack sufficient model quality or materials to recover the specular highlights accurately. Therefore, we propose the LIGHT Specularity Dataset (LIGHTS), constructed from high-quality architectural models with carefully selected lighting and rendering parameters to create a near real-world dataset (see examples in Figure.5.8, 5.9, and 5.10).

The detection of specular highlights requires an understanding of the scene, in its simplest case knowing the camera position, surface geometry, surface material properties and the light source(s) position. Data-driven approaches to the detection of highlights, especially from single-view, fail to encompass this information resulting in a largely color based detection which treats unusual *white* pixels or regions as highlights. While in many cases this assumption holds true, in others it fails to understand cause of the effect and unduly discriminate against white objects. We alternatively treat the scene as 3D and estimate the surface parameters based on the consensus of multiple views that reduces the influence of highlights on the viewer. Thus, we exploit the generalization of single view based methods while incorporating the surface geometry in modeling across views to detect the specularities.

Therefore, the main contribution of this chapter are twofold:

- A new physically based rendering dataset for light analysis from high-quality models with detailed light modeling.
- A multi-view based approach for specular highlight detection which exploits the benefits of single-view techniques and the scene geometry to accurately

detect the specular component in each view.

The chapter is structured as follows, firstly Section 5.1 details the LIGHT Specularity Dataset. Section 5.2 describes a method for the integration of multi-view information for specular highlights detection. Finally extensive evaluation is presented in Section 5.3. We conclude in Section 5.4. Next we review other related work covering both single and multi-view methods as well as relevant datasets.

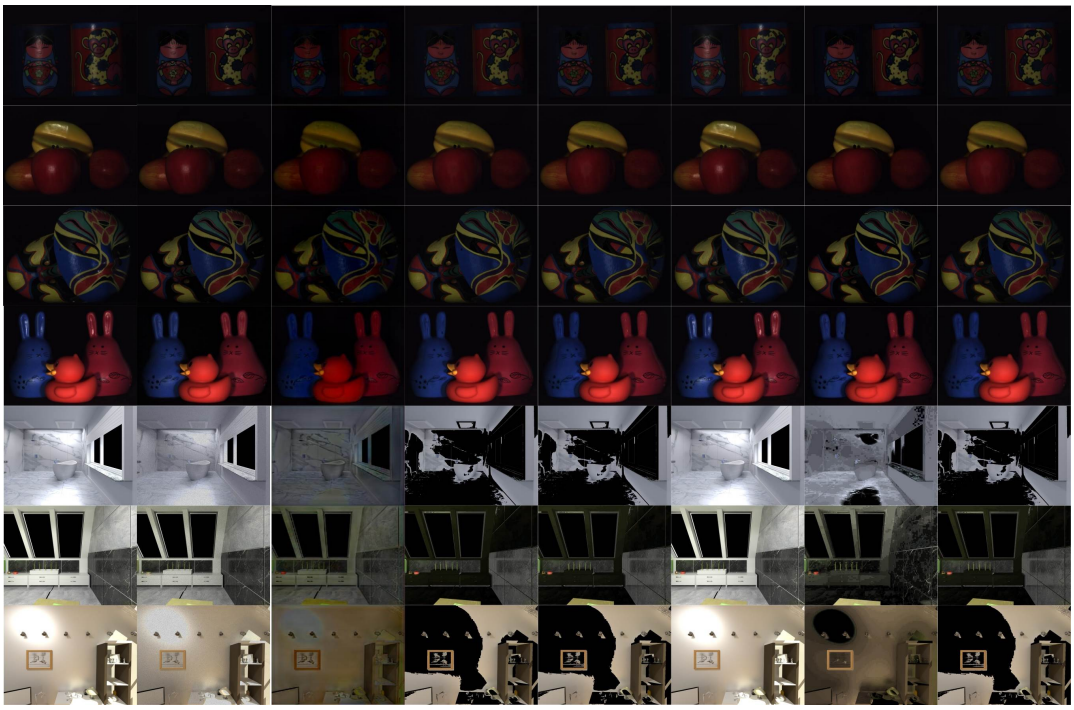


Figure 5.4: From left to right: Original image, Akashi et al [1], Lin et al [2], Souza et al[3], Shen et al [4], Tan et al [5], Yang et al [6], Yamamato et al [7]. The performance of single images specular highlight detection and removal algorithms on conventional images used in evaluation (rows 1-4) which we can refer to it as lab-constrained images, and on normal images (rows 5-7), it is clear that they are performing good on the conventional used images but they have very poor performance on natural (non lab-constrained) images.

5.1 LIGHT Specularity Dataset (LIGHTS)

We propose our LIGHT Specularity Dataset (LIGHTS)[1] for evaluating and comparing techniques related generally to object interaction with light (e.g. specular highlight detection) considering the object material properties as well as their geometry. This dataset provides ground-truth information for diffuse and specular highlights as well as transparency maps, albedo maps, shadow maps, light source intensity, direction and position estimation.

Our dataset is composed of 18 different scenes carefully selected to fulfill Physically Based Rendering (PBR) constraints, and includes models of house rooms. We used PBR scenes as they are structured based on material approximations as near as possible to the physical world resulting in higher accuracy in the ground-truth. For rendering these scenes we used the cycles rendering engine in Blender [127]. We use path tracing for the light transportation model and carefully select the parameters for each light in the collected scenes. In addition we replaced the mesh and environment lights and compensate them with Point, Spot, or Sun lights. Despite blender supporting mesh and environment lights, they are not used in calculating some effects (e.g. shadows), therefore we replaced mesh lights and disabled the "Environment" option so that there is no surrounding environment to emit light from outside to interior of the scene.

Moreover, we intended to have variations in light intensities, so that we can have bright, normal, and low bright scenes. As the scenes were based on architectural designs, these frequently forego walls or ceilings to allow an unobstructed view into the room. Therefore, we added additional geometry to enclose the scene and allow

[1] Dataset: https://pavis.iit.it/datasets/lights

an improved light scattering. Our dataset is organized as following:

- **Camera Information (cam_info)** which is camera related information being the respective intrinsic matrix (K) and Rotation Transformation (RT) matrix. We used the same convention used in Matterport3D [122] and ScanNet3D [123] datasets for these files organiziation which is as in Eq 5.1 for intrinsic matrix where f_x, f_y are the focal length, x_c, y_c are the principal point offset coordinates, and s is the axis skew. And as in Eq 5.2 for Rotation Transformation matrix, while R, and t is the rotation and translation matrices related to the camera orientation and position in the world.

$$\left[\begin{array}{c|c} K & 0 \\ \hline 0 & 1 \end{array}\right] = \left(\begin{array}{ccc|c} f_x & s & x_c & 0 \\ 0 & f_y & y_c & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \tag{5.1}$$

$$\left[\begin{array}{c|c} R & \boldsymbol{t} \\ \hline \mathbf{0} & 1 \end{array}\right] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{5.2}$$

- **Light Information (Light_info)** which is light sources annotation information organized in JSON encoded files. These annotation provides light source type (Point, Sun, or Spot), light source position, light source orientation, light source color, and light source strength. The following JSON sample code in 5.1 is an example of a scene has single light source of type "Spot", while it is objectID is "18", and its object name inside blender file is "cadre deco1" with

the indicated location, orientation, strength, and color.


```
{"lights":
[{
    "Location": [-0.5246, 1.1500, 1.9439],
    "Type": "Spot",
    "IndexOB": 18,
    "Strength": 0.1,
    "Orientation": 128.0,
    "Color": [0.8530, 0.7971, 0.4481, 1.0],
    "Object_name": "cadre deco1"
}]
}
```


- **Mesh Information (Mesh_info)** the complete scene geometry mesh stored as .ply file also in Matterport3D and ScaneNet3D format. We preprocessed the original scenes geometry before using it, we applied decimation so that to reduce the size of mesh but while keeping high level of details for the scene. Later we refined the big faces into small faces so that any face-based operation (e.g. proposed specularity detection technique) has enough number of units which can cover the variations of colors details in objects.

- **Rendering Information (Rendering_info)** the modules which are important to understand how did the light interact with the scene in order to produce the final rendered image. These modules are the Albedo, the Indirect Diffuse, the Direct Diffuse, the Diffuse, the Indirect Specular, the Direct Specular, the Specular, the Indirect Transmission, the Direct Transmission, the Transmission, the Shadow, Depth, Normals, the Object ID, the final rendered image,

and the Lightmap.

The Diffuse, Specular, Transmission modules are the combination of their direct and indirect counterparts according to screen mode as in the Eq 5.3 then multiplied with their color information (each separately) according to the Eq 5.4. While Object ID refers to the objects annotation masks each annotated using its ID. The final rendered image can be calculated from the provided modules as shown in Figure 5.5(a). While the Lightmap is calculated using only light excluding components like color see Figure 5.6(b).

$$Shading = 255 - \frac{(255 - Direct) \times (255 - Indirect)}{255} \tag{5.3}$$

$$E = \frac{Shading \times Color}{255} \tag{5.4}$$

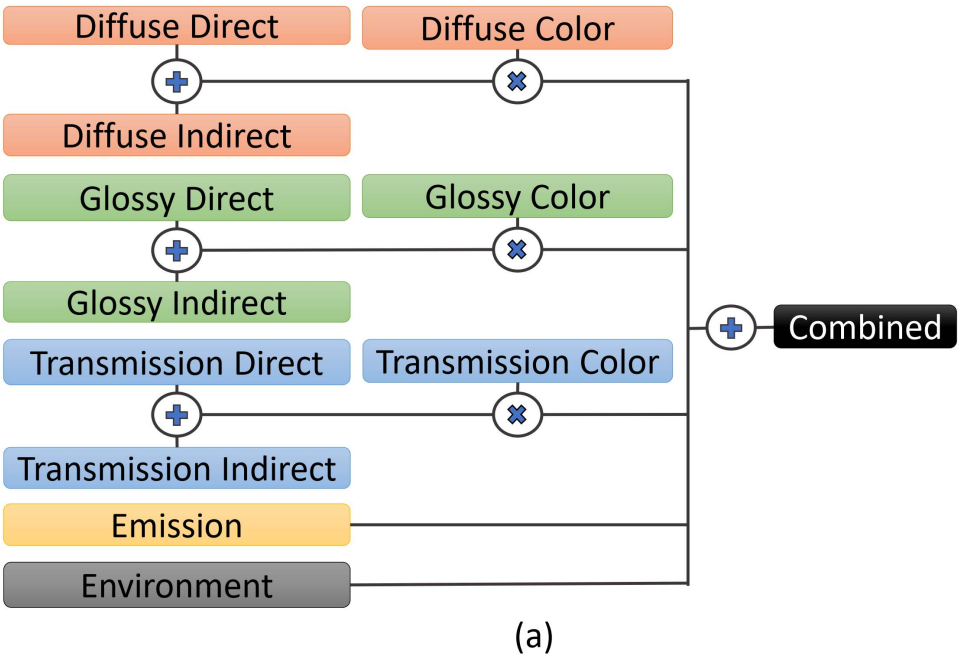


Figure 5.5: We are showing how the rendered image combined in blender from passes to get the rendered image which is represented as (Combined), while on

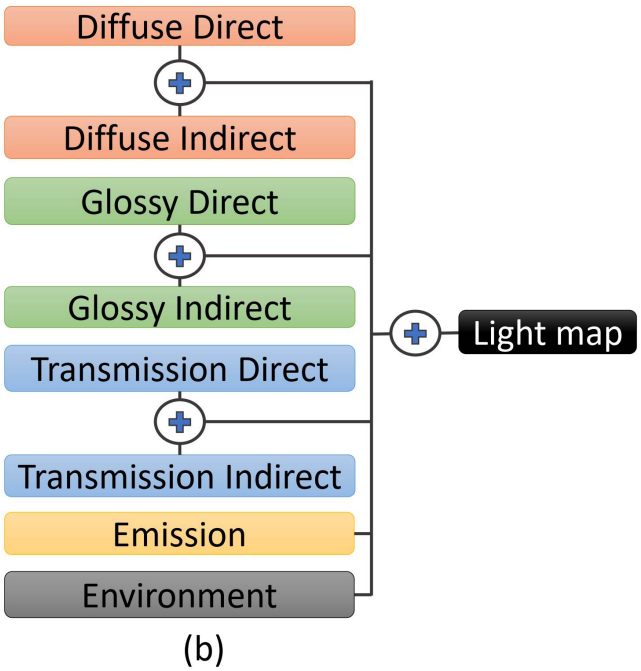


Figure 5.6: We are showing how did we combined the light information from diffuse, specular, transmission, emission, and environment to get our lightmap.

- **Blender Information (Blender_Data)** is our modified ".blend" scenes provided to the community for future use.
- **License Information (licence_info)** is the original designer's licence information.

In the proposed dataset we provide 2,603 rendered views in total using 2,500 samples for rendering, see Figure. 5.7 for distribution over scenes and Figures. 5.8, 5.9, and 5.10 for a sample view from our dataset.

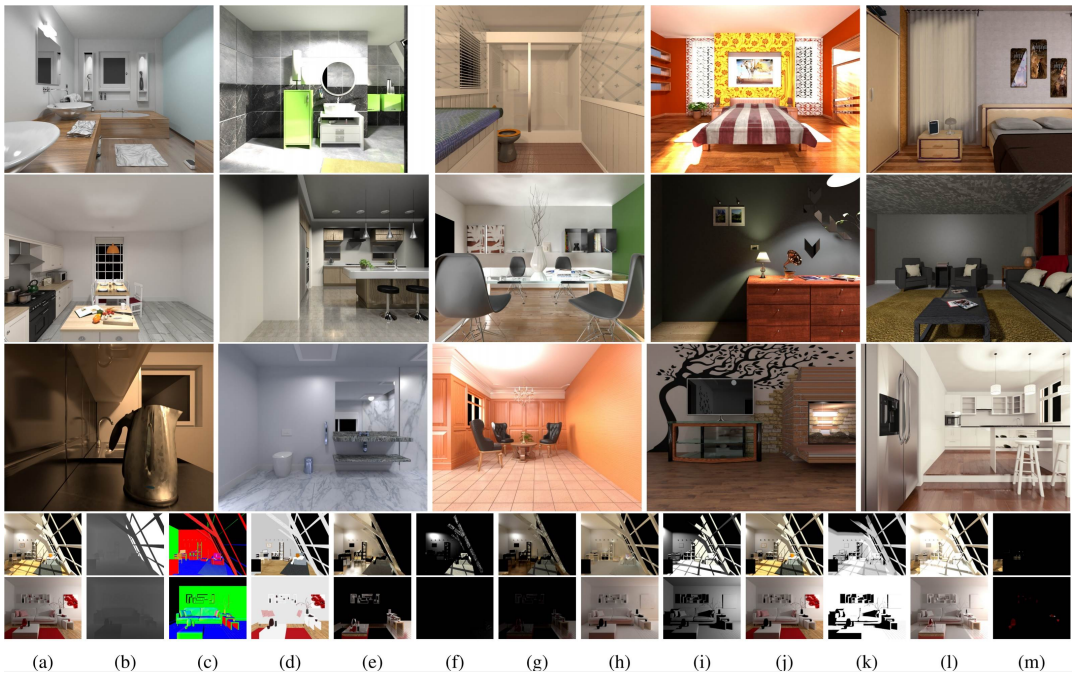


Figure 5.8: Top: Example renderings of some LIGHTS scenes with variation across types of room (kitchens, bedrooms, bathrooms and living-rooms). Bottom: Two examples of the rendered components of the dataset (a) Image (b) Depth (c) Normal (d) Albedo (e) Indirect Specular (f) Direct Specular (g) Specular (h) Indirect Diffuse (i) Direct Diffuse (j) Diffuse (k) Shadow map (l) Light map (m) Transmission.

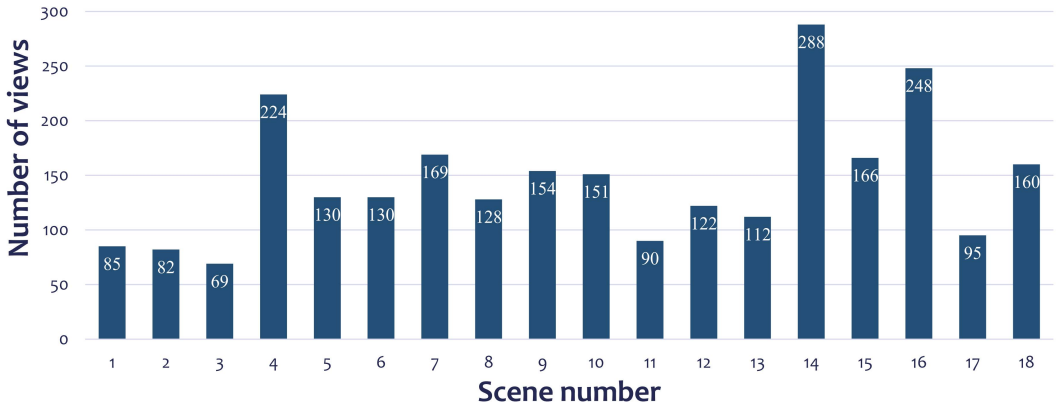
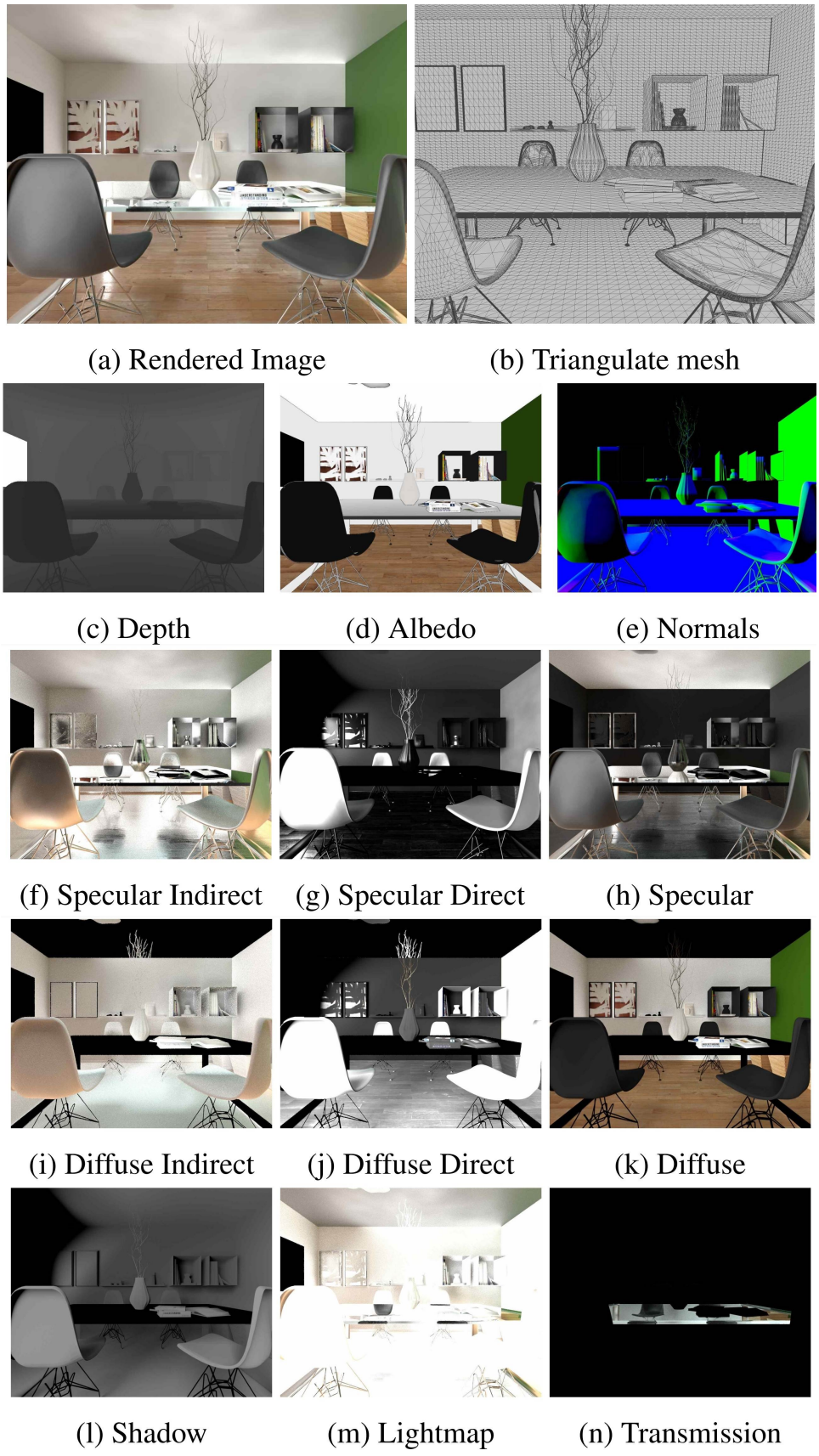


Figure 5.7: The number of views at each scene of LIGHTS dataset



(a) Rendered Image (b) Triangulate mesh

(c) Depth (d) Albedo (e) Normals

(f) Specular Indirect (g) Specular Direct (h) Specular

(i) Diffuse Indirect (j) Diffuse Direct (k) Diffuse

(l) Shadow (m) Lightmap (n) Transmission

Figure 5.9: A sample view from our proposed LIGHTS dataset and its related maps

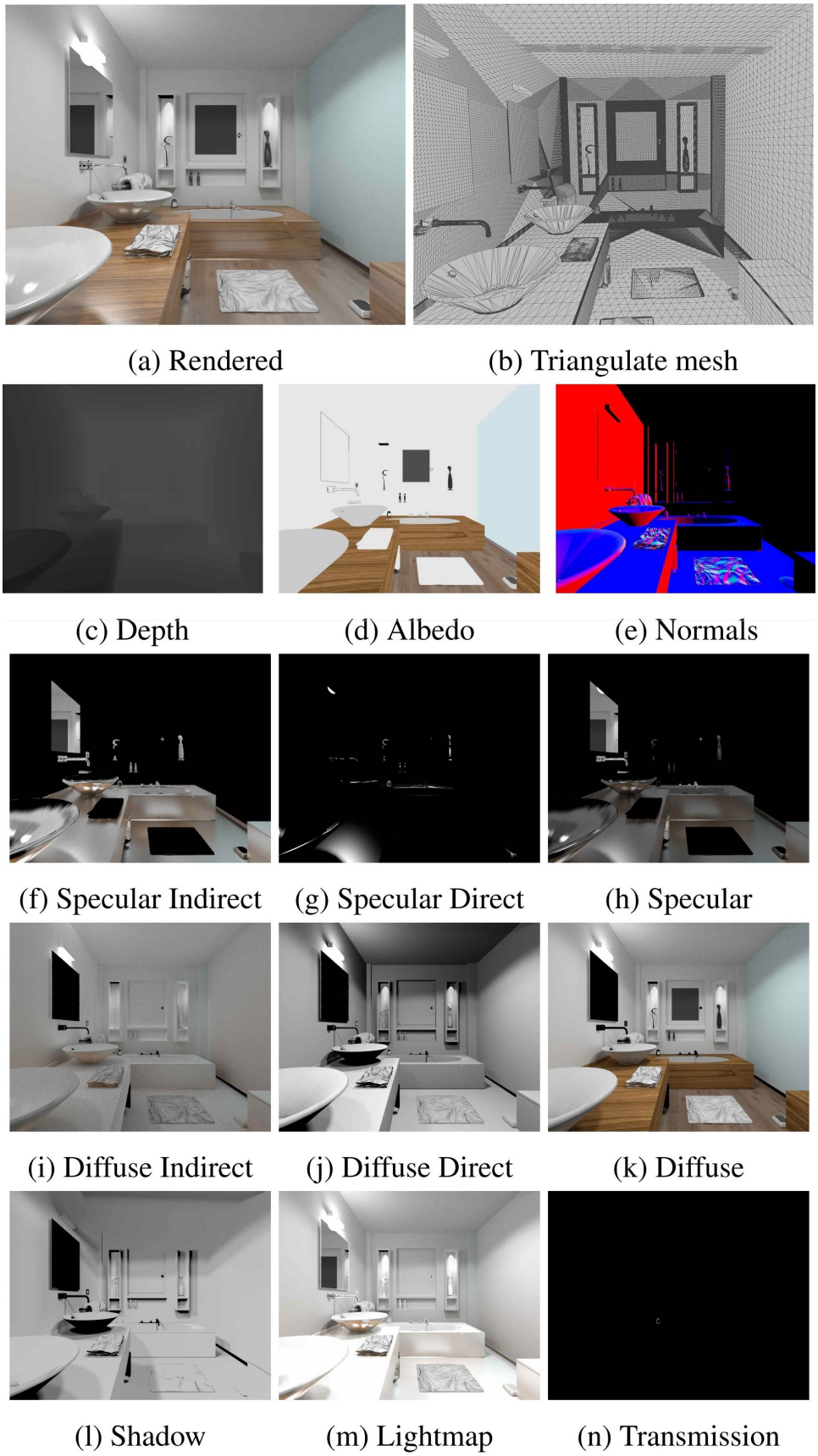


Figure 5.10: A sample view from our proposed LIGHTS dataset and its related maps

5.2 Multi-view specular detection

We propose a multi-view specular highlights detection method which take into consideration the faces of the 3D mesh as seen from multiple rendered images instead of just a single one. Assuming known scene geometry and camera intrinsics and extrinsics, the underlying idea of the approach is that if a single face or group of faces is behaving differently from the surrounding faces in different views, this is an indication that it could be possibly highlighting a specular spot. This is a clear and distinguishable effect as the location of the specular mesh area is highly dependable from viewpoint and light positions. Thus in multi-view, if we remap texture to a common 3D reference model, we would expect strong variations whenever faces have a strong specular presence.

First our approach detects specular regions from single images using an efficient single view specularity detection method [3] to detect specular highlights in all of the scene images and get a mask p for each view. Thus if we have the set of images in the scene $I = \{i_0, ..., i_N\}$, the set of faces in this scene is $F = \{f_0, ..., f_O\}$, and the set pixels within each I image is $H = \{h_0, ..., h_x\}$. For each mask p_N detected from an image i_N, the projection of face f_O in the mask p_N or the image i_N will be called f_{NO}. For each mask p_N, its corresponding faces f_{NO} will be divided into specularity potentials s or diffuse potentials d according to Eq. 5.5, while all calculations are based on L channel from L*a*b color space.

$$f_{Nz} = \begin{cases} s, & \text{if } \lfloor \frac{\sum_{x=1}^{n} p_N(h_{NOx})}{n} \rceil = 1 \\ d, & \text{otherwise} \end{cases} \tag{5.5}$$

where n is the total number of pixels inside the face f_{NO}, and $\lfloor \rceil$ defines the mathe-

matical rounding to the nearest integer. This means that at each face f_{NO} the decision will be based on the majority of its corresponding pixels h_{NO} inside the mask p_N.

Thereafter, for each single view N, we have to iterate through all the other views J to decide whether the current specular potential faces in view N are specular or not based on their intensity on all other views. This decision is based on a adaptively calculated threshold between the two views compared each time. The threshold calculation is based on the common diffuse potentials between the two views by using Eq. 5.6 such as

$$t = d_{mean} + \phi \times s_{mean} \tag{5.6}$$

where d_{mean} is the mean value of common diffuse-potential faces luminance between the two frames, and so s_{mean} is the mean value of common specular-potential faces. While ϕ is a constant which control the sharpness of detected specularities because it is controlling the weight of specular part in the threshold equation.

During each iteration, faces that do not meet the threshold criteria are excluded from potential specular faces s as shown in Eq. 5.7:

$$w(f_{NO}) = \begin{cases} keep\ in\ s, & \text{if } (f_{NO} - f_{JO}) \geq t \\ exclude\ from\ s, & \text{otherwise} \end{cases} \tag{5.7}$$

where f_{NO}, and f_{JO} are specular-potential face luminance value in frame N and its correspondence face luminance value in frame J respectively. Finally, our detected specular faces will be the rest (non excluded) faces. Our proposed algorithm steps is as shown in Algorithm 1.

Algorithm 1: Multi-view specular detection

Result: Specular faces at each view

```
Calculate potential specularity mask p using a single image detector;
for each frame N do
    for each face visible in N do
        use the mask p to mark faces as specular s or diffuse d using Eq. 5.5.
    end
    for each frame J different from N do
        if exist common d faces with frame N then
            calculate threshold t using Eq. 5.6;
            exclude faces from s using Eq. 5.7;
        else
            skip frame;
        end
    end
    specular faces in frame N=s;
end
```

5.3 Experimental Results and Analysis

In this section we are going to show the evaluation of our proposed technique against the multi-view technique of Lin et al. [12] and show difference of accuracy achieved over the used single image detection technique of Souza et al [3]. We compare the efficiency of algorithms from the processing time and accuracy quantitative metric points of views.

5.3.1 Evaluation metric

The output from our technique is a binary mask indicating the visible specular highlights detected by our algorithms, while the ground truth specular map contain gray values representing how strong is the specular component at each pixel. To compare these two types of images we need to threshold the ground truth map to be in the same form like the estimated mask, but the question is, which threshold value we should use? The solution to this problem should not be limited to a specific threshold value as the strength from the specular component varies as a result of the materials and the objects color. This should allow us to calculate the error based on the threshold which maximize the similarity between the estimated mask and the thresholded ground truth specular map. Our solution to this problem is to threshold the ground truth map using a range of thresholds one at a time, and then for each one, we calculate the error between it and our estimated mask using IOU (Intersection Over Union), in this way we will have multiple accuracy values (as many as the threshold values). The final adopted accuracy is calculated based on all calculated IOU accuracies from all threshold values. The ground-truth specular map is in RGB format, at first we had to convert it to L*a*b color space to use the luminance channel L as our gray level ground truth map G_M. At the end, the overall accuracy for

number of views can be calculated as the average of all calculated accuracies across all viewpoints. Given that S_T is the set of used ranges in the ground truth specular map, and the ground truth specular map G_M, the thresholded ground truth mask G_T can be calculated using Eq 5.8:

$$G_T = G_M > T, \quad \forall T \in S_T \tag{5.8}$$

Then the Accuracy A value between a single estimated mask M and G_T can be calculated using the Eq 5.9 while TN is the total number of used threshold ranges.

$$A = \frac{\sum \frac{M \cap G_T}{M \cup G_T}}{TN}, \quad \forall T \in S_T \tag{5.9}$$

At the end, the overall scene or group of views accuracies A_T can be calculated for all views within the set of all views of that scene $I = \{i_1, i_2, i_3, ..., i_N\}$ using the Eq 5.10

$$A_T = \frac{\sum_{m=1}^{N} A_m}{N} \tag{5.10}$$

5.3.2 Environment Setup

We tested single-view detector algorithms on several images from previous works and also on our synthetic dataset images as shown in Figure. 5.4. Multi-view techniques like ours and baseline technique of Lin et al. [12] evaluated only on our LIGHTs synthetic dataset.

Regard single images detection and removal techniques like Souza et al [3], Lin et al[2], and Yamamoto et al [7] we used the authors source code. On the other

hand for Yang et al [6], Shen et al [4], Tan et al [5], Akashi et al [1], we used the matlab version provided by Yamamoto et al [7]. In addition to implementing the proposal of Lin et al. [12] as mentioned in their work. For a fair comparison between our technique and baseline technique of Lin et al. [12] as they assumed unknown geometry and camera intrinsics and extrinsics, we used the geometry and camera information for their technique, in other words we considered that these information is known for their technique and did not reconstruct geometry and find the pixel correspondences between the images, but directly use the accurate pixels correspondences from the ground-truth geometry and projection matrices and use them in baseline evaluation.

As stated in baseline technique [12], we grouped our multi-view images in triplets (right, center, left), while for each center view we iterate through all possible combinations of left and right views, later we calculate the multi-CHD for each center view by calculating tri-CHD for all triplets related to this center-view, finally we set the multi-CHD voting threshold of 50%. The baseline technique authors evaluated it on their synthetic multi-view images while the distance between consecutive views is fixed, while our multi-view images are sparse. In this chapter we evaluate the baseline efficiency for real scenario multi-view scenes.

For our technique settings, we used the value of $K = 0.5$ for Eq. 5.6, while decreasing its value mean increasing detected specularities, and vice versa. While we evaluated using the last 100 range in specular map mean that S_T will be in the range of (156 to 255), as it does not make sense to check specularities in the lower ranges as it does not appear on it.

5.3.3 Results and analysis

We evaluated our multi-view proposed technique against Lin et al. [12] and the used single image detector of Lin et al. [12] on the proposed LIGHT Specularity multi-view dataset from speed and accuracy points of view.

The provided timing on Table 5.1 is using Matlab on headless Ubuntu devices, each has (56-core) with 128GB memory, it is showing the processing time for each technique on some/all images from scenes alongside with the total number of views at each scene. From the utilized times we can notice that the Lin et al. [12] is time consuming technique, this is because the number of views on the scenes is not the only controller of the processing time for both algorithms, but implicitly there are also the used unit in calculation (faces in our case, and pixels in Lin et al. [12] case), the number of matches between views at each iteration, and the used way from each technique to iterate through the images (ours is using the whole image at once, while Lin et al. [12] is iterating row by row), while the resolution of our dataset is 1024x1280 which was a processing exhaustive for the baseline technique. We have to mention also that both techniques have to iterate over all of views to calculate the specularities in a single view.

On Table 5.2 we are showing the evaluation accuracy of our proposed technique against the Lin et al. [12] technique against the single image detector technique of Souza et al[3] which used by our technique. The first column of the table is showing the result of the three algorithms on **336** views from multiple scenes, while the second column is evaluated on all of our dataset views which is **2603** view. Despite that we were not able to evaluate the Lin et al. [12] technique on the full dataset due to its processing time, but **336** is at least 10x more evaluation images compared to previous number of evaluation images in recent works. Our algorithm achieved

higher accuracy than [12] and [3] by 3.5% and 1.3% respectively. Our approach also proved to be faster w.r.t. Lin et al [12] by approx. 4k times in average (See Table 5.1 for full performance comparison). We also show a qualitative comparison in Fig. 5.11, Fig. 5.12, and Fig. 5.13 against both [12, 3] where it can be seen that our approach outperforms the other approaches, especially on white objects or saturated areas which can be easily confused for specularities. The use of faces instead of pixels in our framework has two advantages, 1) it is more robust to noise relying on the consensus across the face, and 2) using faces capitalizes on speed compared to processing individual pixels.

Figure 5.11: From left to right: rendered image, specular map, our estimation, Lin et al. [12] estimation, single view estimation (Souza et al[3]).

On the other hand, face size is controlling the accuracy, faces should be big as possible to speed the process but at the same time should be sure that each face pixels mostly have the same behavior or properties, in other words all of them should be small enough to hold either specular or diffuse, but not mix specular and diffuse in the same face. this mean that we have to compromise between speed and accuracy.

We show some qualitative evaluation of our technique against Lin et al. [12], Souza

et al[3] techniques in Figures. 5.11, 5.12, and 5.13, please refer to electronic version for high quality. As shown in the figures, multi-view techniques did not detect white objects as specularites on contrast to the single view technique. Our technique is showing a satisfying results in compared to compared techniques.

Table 5.1: Processing time (in sec) of the two compared techniques per number of views used. In the last row, the average time consumed for all the scenes per method.

Scene ID	Total views	Ours (Time/#views)	Lin [12] (Time/#views)	Ours (Time/view)	Lin [12] (Time/view)
1	85	347/85	1982k/85	4.02/1	23k/1
2	82	177/82	1549k/82	2.15/1	18.9k/1
3	69	104/69	1234k/69	1.49/1	17.8k/1
4	224	377/224	993k/5	155/1	198k/1
5	130	504/130	493k/19	3.88/1	26k/1
6	130	674/130	162k/6	5.19/1	27k/1
7	169	1.2k/169	28k/1	7.31/1	28k/1
8	128	541/128	157k/10	4.23/1	15.7k/1
9	154	161/154	248k/10	1.05/1	24.8k/1
10	151	851/151	242k/10	5.64/1	24.3k/1
11	90	263/90	158k/11	2.92/1	14.4k/1
12	122	480/122	45.3k/4	3.93/1	11.3k/1
13	112	442/112	259k/10	3.95/1	25.9k/1
14	288	3k/288	220k/1	10.5/1	220k/1
15	166	647/166	28.4k/1	3.9/1	28.4k/1
16	248	1.95k/248	206k/1	7.88/1	206k/1
17	95	258/95	173k/10	2.72/1	17.3k/1
18	160	418/160	28k/1	2.61/1	28k/1
Avg.	145	687	456K	12.69	53.04k

Table 5.2: Accuracy evaluation using proposed metric A_T, two columns are presented, the first one obtained by evaluating the techniques on 336 image from multi-scenes, while the second column is by evaluating on the full LIGHTs dataset.

	A_T**/#Views**	A_T**/#Views**
Single-view [3]	0.0282/336	0.031/2603
Multi-view [12]	0.0146/336	-
Multi-view (Ours)	0.0502/336	0.04/2603

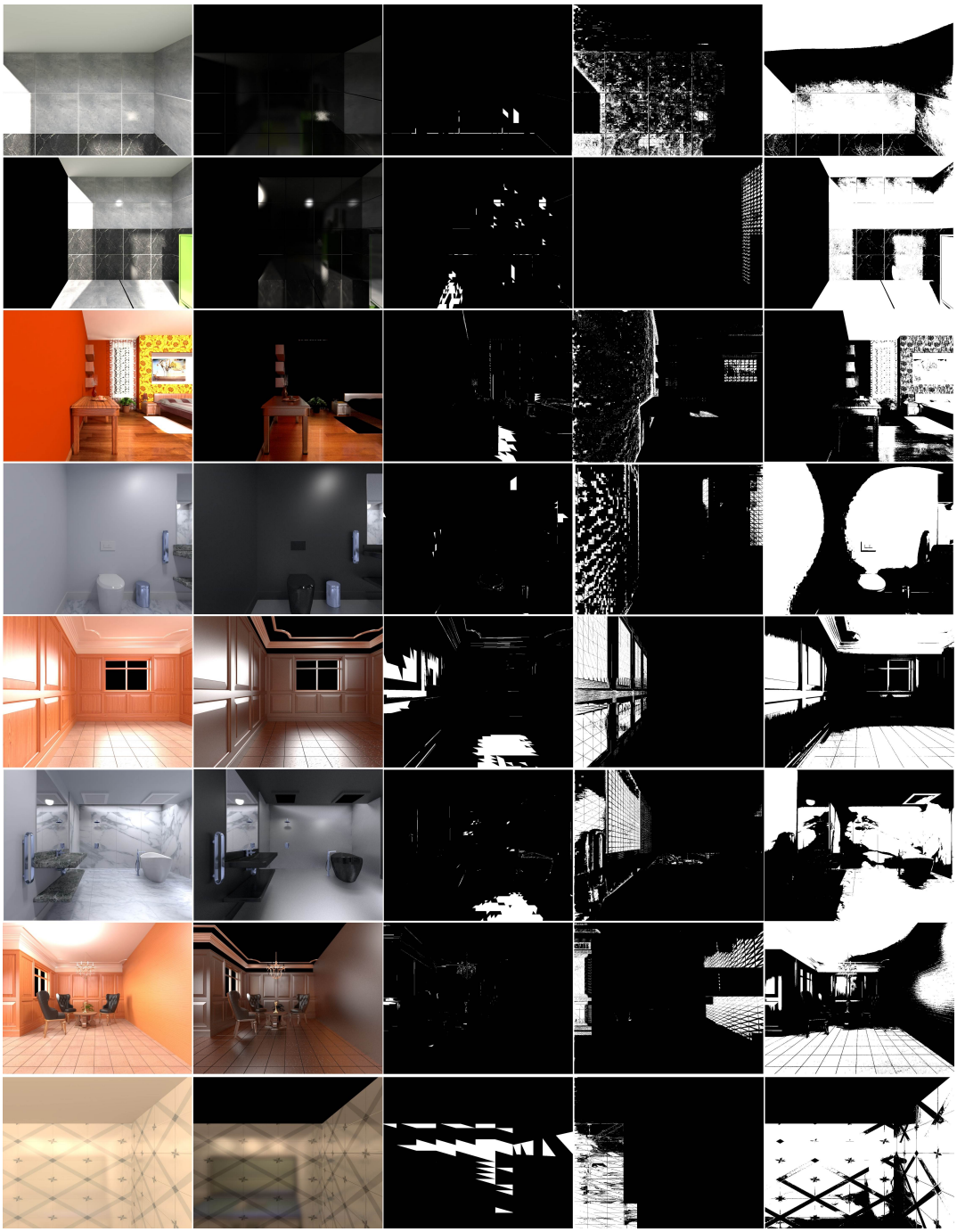


Figure 5.12: From left to right: rendered image, specular map, our estimation, Lin et al. [12] estimation, and our dependant single view estimation (Souza et al[3]).

Figure 5.13: From left to right: rendered image, specular map, our estimation, Lin et al. [12] estimation, and our dependant single view estimation (Souza et al[3]).

5.4 Conclusion

We have presented an effective multi-view image specular detection algorithm. The proposed algorithm extends previous works on single image specular detection and removal which employed a data-driven approach to detect specular highlights. By aggregating the extracted information from multi-view images, we were able to distinguish between specular regions and non-specular regions which confuse single image methods. The proposed method widen the future research of specular removal to deep models that can be trained on pre-processed multi-view natural images common during reconstruction capture. Our method accuracy is limited by faces size and the number of faces in the scene, despite that, it is still faster than pixel based methods. While we found that the traditional used sets of evaluation images were not reflecting the real performance of detection algorithms specially on the natural images, we presented a physically based rendering LIGHT Specularity Dataset (LIGHTS), which simulating the real light transportation, material properties as possible to be used for light analysis evaluations. The dataset contains wide variations of light setup, materials, and objects. In future work the method can be applied to large scale multi-view datasets (e.g. Matterport3D and Scannet), or incorporating semantic information to train deep models to overcome confusion between specularities and similar effects on single images.

Chapter 6

Multi-View Mesh Colors

We address the issue of creating a consistent mesh texture maps captured from color uncalibrated scenes which will produce an improved scene visualization and understanding. We compute a color prior from the cross-correlation of observable view faces and the faces per view to identify an optimal per-face color. We then use this color in a re-weighting ratio for the best-view texture, where the best-view texture is identified by prior mesh texturing work [13, 14], to create a spatial consistent texture map. Despite our method not explicitly handling spatial consistency, our results show qualitatively more consistent results than other state-of-the-art techniques while being computationally more efficient. We evaluate on prior datasets and additionally Matterport3D showing qualitative improvements.

The objective of 3D Reconstruction is to achieve an accurate digital 3D mesh representation of the real world in terms of precision of the vertices and their respective colors or texture. However, techniques have improved in accuracy for the vertex estimation of the 3D mesh with improved sensor information or fusion techniques, but providing a consistent color or texture map across the mesh remains challenging. Otherwise hyper sensors have to be used during capturing. The optimal mesh color and texture will be firstly accurate – a challenge in an uncalibrated environment; and secondly consistent – without disjoint patterns occurring across the faces. Such objectives are challenging as during capture lighting and other environmental

factors are difficult to control which frequently create artifacts in the reconstruction. Providing such an accurate mesh allows for relighting of the scene useful in a variety of applications. We focus on the latter issue of removing disjoint patterns and argue that the answer is within the views if aggregated correctly. Considering diffuse scene, we depend on the inter and intra views relation between colors as a cross-correlation factor and in turn can be used to optimize a texture map creating a consistent texturing across the surface. Neglecting such a relation results in inconsistent colors from multi-view even using averaging, or prior mesh texturing methods (see Fig 6.1). For this reason, even the State-of-Art mesh texturing algorithms like [14] still suffering from this inconsistency in their texture, which can be improved through our aggregation. In this chapter, we present a relatively simple technique to provide spatially consistent texturing over a mesh. We achieve this by using face-based colors extracted from their respective images as well as faces texture maps. At first, we estimate consistent faces colors by infilling across views then using trimmed mean to aggregate information which, in essence, this re-weights the color distribution based on our cross-correlation factor. We then use the ratio of our consistent face color to the texture mean to optimize the texture images from prior mesh texturing approaches creating a consistent texture map. We can list this chapter contributions as:

- Estimating consistent per-face colors;
- Refinement method for prior mesh textures by correcting the textures using estimated per-face colors.

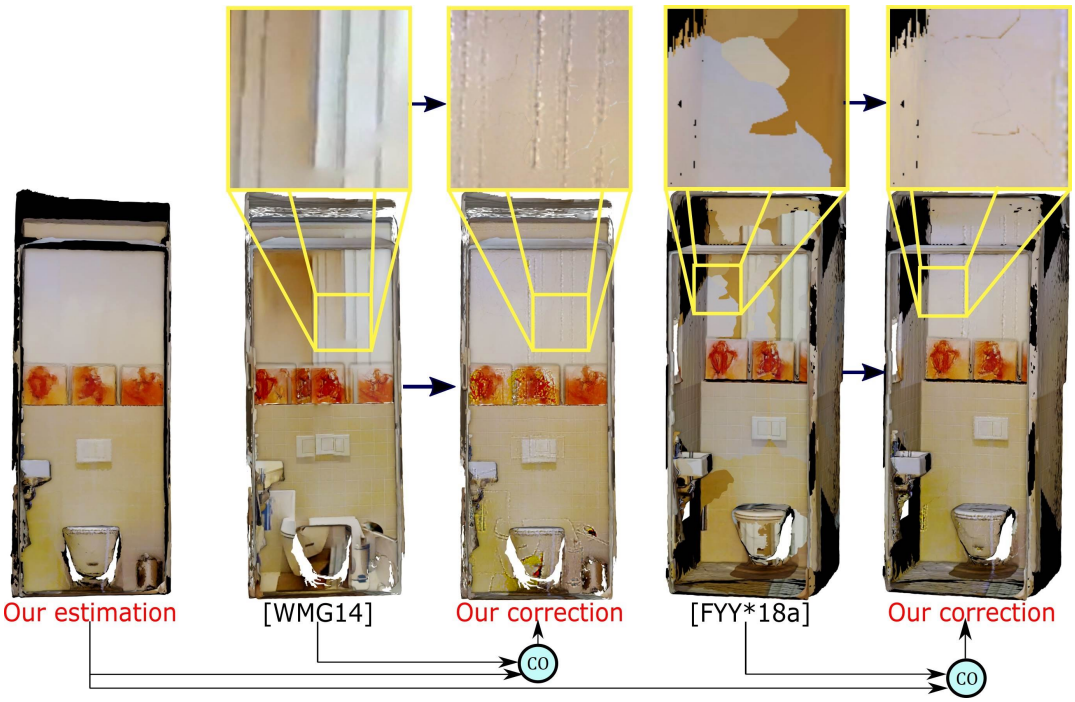


Figure 6.1: Our per-face colors estimations as low resolution texturing (column 1), and our corrections to high-resolution textures generated from state of art techniques [13, 14] tested on Matterport3D [9] scene, while "CO" refers to correction. Our results showing better and competitive visual quality.

6.1 Consistent Mesh Colors

We are trying to provide consistent per-face mesh colors (one color per face, also called seamless) and therefore using it to correct texture images and provide consistent colors texturing. Our input is multi-view data and its reconstructed mesh alongside with its calibrated camera intrinsic information see Figure 6.2 for the full system diagram. We are infilling the unobserved mesh faces colors at each view based on the overlap between multiple views and the relations between mesh faces within the same view in 6.1.1. This infilling will happen for each view to put each unobserved mesh face under the same capturing environment of observed mesh faces in the same view. After infilling all unobserved faces at all views, the final consistent

color value for each face is estimated through aggregation over all infilled views in 6.1.1. Finally, we use the estimated colors to provide consistent color mesh texturing by correcting a previously textured mesh in 6.1.2.

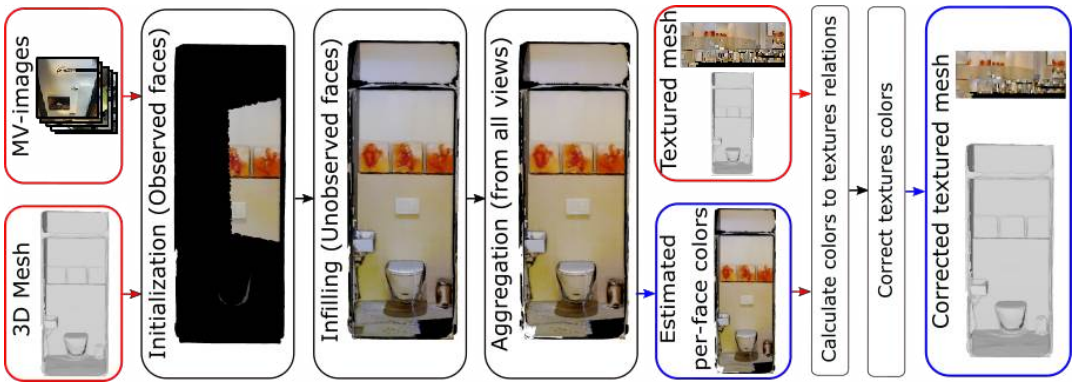


Figure 6.2: Red, blue, and black arrows provide input, output, and transferring to the next step respectively. Using multi-view images and its reconstructed 3D-mesh we estimate one color per face, then use it alongside texture images to provide corrected color textured mesh.

6.1.1 Estimating per face colors

In this section, we are presenting a per face consistent color estimation algorithm, in this step we are not trying to do mesh texturing, although it could be used as a low resolution (per face) mesh texturing in which only one color is assigned to the face, but our concern here is to provide a consistent color distribution over mesh faces, which are not affected by the input images different factors from different view angles. Firstly we will use the cross-correlation between overlapped views (common faces) to understand how to simulate the change in color degrees from one view to another, we are referring to this relationship as “inter-view” relation . Then, within each single view, we assume that cross-correlation between neighbor faces remain nearly the same on the other views even in different scales, we are referring to this relationship as "intra-view" relation which is a relation between different

faces estimated within single view then we use it in another view.

For mathematical simplicity, we assume, that our images are gray-scale. Later, we will extend to RGB colored image. by calculating all steps for each channel separately. Given a set of N views $I = \{i_1, i_2, i_3, \ldots., i_N\}$, and a 3D mesh containing the F faces set, while $F = \{f_1, f_2, f_3, \ \ldots, f_O\}$ and O is the total number of faces in the mesh.

Frames have some observed faces on it $f_v = \{f_{v_1}, f_{v_2}, f_{v_3}, \ldots., f_{v_N}\}$, where v stands for $observed$. These faces has "inter-views" relations between each other, in other words there are some overlapped faces across views. Also, frames have unobserved faces (either outside viewport or occluded) $f_n = \{f_{n_1}, f_{n_2}, f_{n_3}, \ldots., f_{n_N}\}$ where n stands for $unobserved$. The set $F = [f_v, f_n]$ for all frames compose the full set of faces in the mesh. As initialization, for a specific face we assume that the majority of views see its correct color, while the minority see wrong colors and those are much less correlated. Based on this assumption we assign the initial colors for observed faces at each frame separately from its corresponding frame pixels. But we are not using the Mean which is the common way to do this as it is too sensitive to extreme observations, but, we are using Trimmed Mean see Eq. 6.1 which is designed to solve that problem. which can be defined as the following, given a g number of observations, and the d_e is the observed values for which we would like to find the trimmed mean. We reorder them in ascending order, then compute $K = g\alpha$. While α is half of the percent of trimming. e.g. to simulate the Mean we use $\alpha = 0\%$ Trimmed Mean. We use $K = g\alpha$ to trim k observations at both ends while $g\alpha$ should be an integer corresponding to the number of trimmed observations. While R is the remaining number of observations $R = g - 2K = g(1 - 2\alpha)$.

$$T = \frac{1}{\lfloor R \rfloor} \times \sum_{\lfloor K \rfloor + 1}^{g - \lfloor K \rfloor} d_e \tag{6.1}$$

To simulate the changing in color degrees from a view e.g. (b) to another e.g. (j), we will depend on the overlapped faces between current frame b and the other frame j which can be written as Eq. 6.2

$$f_{O_{bj}} = f_{v_b} \cap f_{v_j} \quad \forall\, b \neq j, \quad b, j \in N \tag{6.2}$$

where O stands for overlapping. We will refer to the correspondences of unobserved faces from frame b in observed faces on frame j as $f_{v_j}(f_{n_b})$, on the same manner $f_{v_b}(f_{O_{bj}})$, $f_{v_j}(f_{O_{bj}})$ refers to the correspondence of overlapped faces in observed faces in frames $b\ and\ j$ respectively. The inter-views relation $Finter_{bj}$ between faces from frames b and j can be formulated as Eq. 6.3.

$$Finter_{bj} = \frac{1}{N-1} \sum \frac{f_{v_b}\left(f_{O_{bj}}\right)}{f_{v_j}\left(f_{O_{bj}}\right)} \tag{6.3}$$

Now we can use $Finter_{bj}$ to change some faces colors from frame j domain to frames i domain, we will refer to it as intermediate estimations. e.g. we can change some faces color values $P_{V_j}(P_{N_b})$ from frame j to agree with color values in frame b as $\hat{P}_{N_{bj}}$ as shown in Eq. 6.4. We are doing this for calculating what would be the color values of the unobserved faces if they were observed in this frame. This operation is called "Infilling".

$$\hat{f}_{n_{bj}} = f_{v_j}\left(f_{n_b}\right) Finter_{bj}. \tag{6.4}$$

Figure 6.3: Mesh infilling: Two input views for the same scene, observed faces, and the result of infilling. The visible part of the scene is enclosed in the red rectangle. It is clear that there is a difference between infilled faces color degrees from the two views. Note: the black faces in the infilled part is for the faces which are not appearing from any view.

This last equation will be repeated between the desired frame b and all other frames to have all possible intermediate estimations for only the unobserved faces in frame b. Note that the intra-view relation implicitly encoded in the last equation, as we use the relation between unobserved faces in the current frame with the observed faces in the same frame. The final estimated value for each unobserved face m from f_{n_b} for frame b could be estimated using simple averaging as shown in Eq. 6.5.

$$\hat{f}_{n_{bm}} = \frac{1}{h}\left(\hat{f}_{n_{bj}}\right) \forall\, j \neq b,\ m = 1:h \tag{6.5}$$

where h is the number of intermediate estimations for the single face m from other frames, (note that h is varying for each face, as face will have an intermediate estimation from frame j only in the case that this face is observed on it). This operation will yield $\hat{f}_{n_b}$ which is the set of all estimated unobserved faces for frame b. The full set of mesh faces for frame b will be $\hat{Q}_b = \{f_{v_b}, \hat{f}_{n_b}\}$. By now, we have the full set of colors for all faces (either observed or unobserved) for a single view. We will repeat the same operations for all views to get the full set of colors for all faces from each viewpoint $\hat{Q} = f(\{\hat{Q}_1, \hat{Q}_2, \hat{Q}_3, \ \ldots, \hat{Q}_O\})$. Finally, to get the final consistent color for each face, we will use the Trimmed Mean as the aggregation function between

different views colors which are currently represented in $\hat{Q}$ sets. The output from aggregation process is our final estimations which is a single color value per face to form the set of colored faces $\hat{f} = \{\hat{f}_1, \hat{f}_2, \hat{f}_3, \ \ldots, \hat{f_O}\}$. For RGB colored image all steps will be calculated for each channel separately.

6.1.2 Correcting mesh texturing

The purpose of this step is to overcome the limitation of mesh resolution for the estimated per-face colors and provide high resolution color consistent mesh texturing, we are doing this by using the colors from estimated per-face colors and the detailed texture pattern from the textures to produce the consistent color textures. As known, each face O has a corresponding texture image. Furthermore, the face color should be corresponding to the mean value of its texture pixels colors, based on this we are aiming to minimize the difference between the estimated faces colors and the mean value of faces textures pixels values M. This difference will be minimized through correcting the texture colors for each face O by calculating its new pixels values $\hat{x}_O$ using Eq. 6.6 based on old pixel value x_O, estimated per face colors $\hat{f}_O$, and the mean value of face textures M_O.

$$\hat{x}_O = x_O \frac{f_O}{M_O} \tag{6.6}$$

In this process, we can correct any textured scene by using its estimated per-face colors.

6.2 Experimental Results and Analysis

We evaluated our proposed texture correction for the output from Waechter et al [13], and Fu et al. [14] algorithms against the original not corrected output from Waechter et al [13], Sai et al.[104] and Fu et al. [14] algorithms by the mean of projected textures. Also, we are showing the difference between our face-based estimated colors against using averaging. We evaluated on a variety of scenes from different datasets: 1)Matterport3D [9], 2) Scenes from [128] , and 3) Scenes from [15] which is available on [129]. The Matterport3D dataset views have been captured from different random locations and different distances offering a variety of scales, while in scenes provided by [14] and from [15] the viewpoints are dense (video-like) and not offer such challenging variations. We used the authors code for Fu et al. [14], and Waechter et al [13]. Regard Sai et al.[104] we used the available implementation from [130] . We used a minimal number of images in scenes from [15] scenes in our tests (e.g. 300 images instead of 3000 in lounge), also we simplified the scene and kept 20% of its original mesh data to limit the computations to hundreds of views instead of thousands in all of our tests. For the trimmed mean, we are using the value of α as 30%. For Sai et al.[104] we used 10 scales and 50 iterations of alignment per scale. Unfortunately, the algorithm is not providing the output as a 3D textured model, but as a projected images without mapping information between faces and texture images, so we could not correct it.

We show the qualitative evaluation results in Figure 6.4. We are using some red marks **A, B, C, D, E** for analysis purposes. Marks **A** and **B** shows a clear inconsistency between neighbor regions caused by texturing or averaging from multi-views while it does not exist in their correspondences in the original images. Our results in the columns (3, 7, 8) are clearly showing that we successfully overcome this prob-

lem in the estimated face-based colors, and in corrected texture models. Also, we can generally notice that textures from Waechter et al. [13] are not perfectly registered on geometry and sometimes gives the effect of duplication as in **D** marks. Our algorithm hid most of it and provided textures more visually closer to the original, but still, there is inappropriate texture registration behind our colors which sometimes cause undesired patterns like in **E** marks. Regard our algorithm we can notice that it sometimes provide brighter results than the original image e.g rows (5, 7, 8, 9) from up, and sometimes provide darker e.g. (row 6) this is because we aggregate from multi-views images, some of them are darker and some of them are brighter, so comparing our results to some of them could be darker than the original views, and at the same time, it is brighter than other views. Moreover, our estimation brightness could be adjusted after aggregation without affecting consistency between colors or losing color degrees because of trimming to serve for different applications. We can also notice that there are some visible seams in our corrected textures in **C** marks, this is because hiding seams require adding margins to the textures, that is what Waechter et al [13], Fu et al. [14] are doing, but during our correction process we only have the output from both algorithms which contain the mapping information between mesh triangles and textures but not for the margins, therefore the margins have not been corrected causing seams to be visible. This could be solved by providing this information or by using our approach during the texturing process. Despite we based our assumption on having diffuse objects, but still our technique is providing good results from specular objects as shown on the chair and ball on the fourth and last row in fig 6.4 respectively, the credit for this is for using trimmed mean in aggregation from all views. No doubt that increasing the number of mesh faces, will provide our technique with additional benefit by providing more precise face-based colors which in turn used for providing extra sharp corrections.

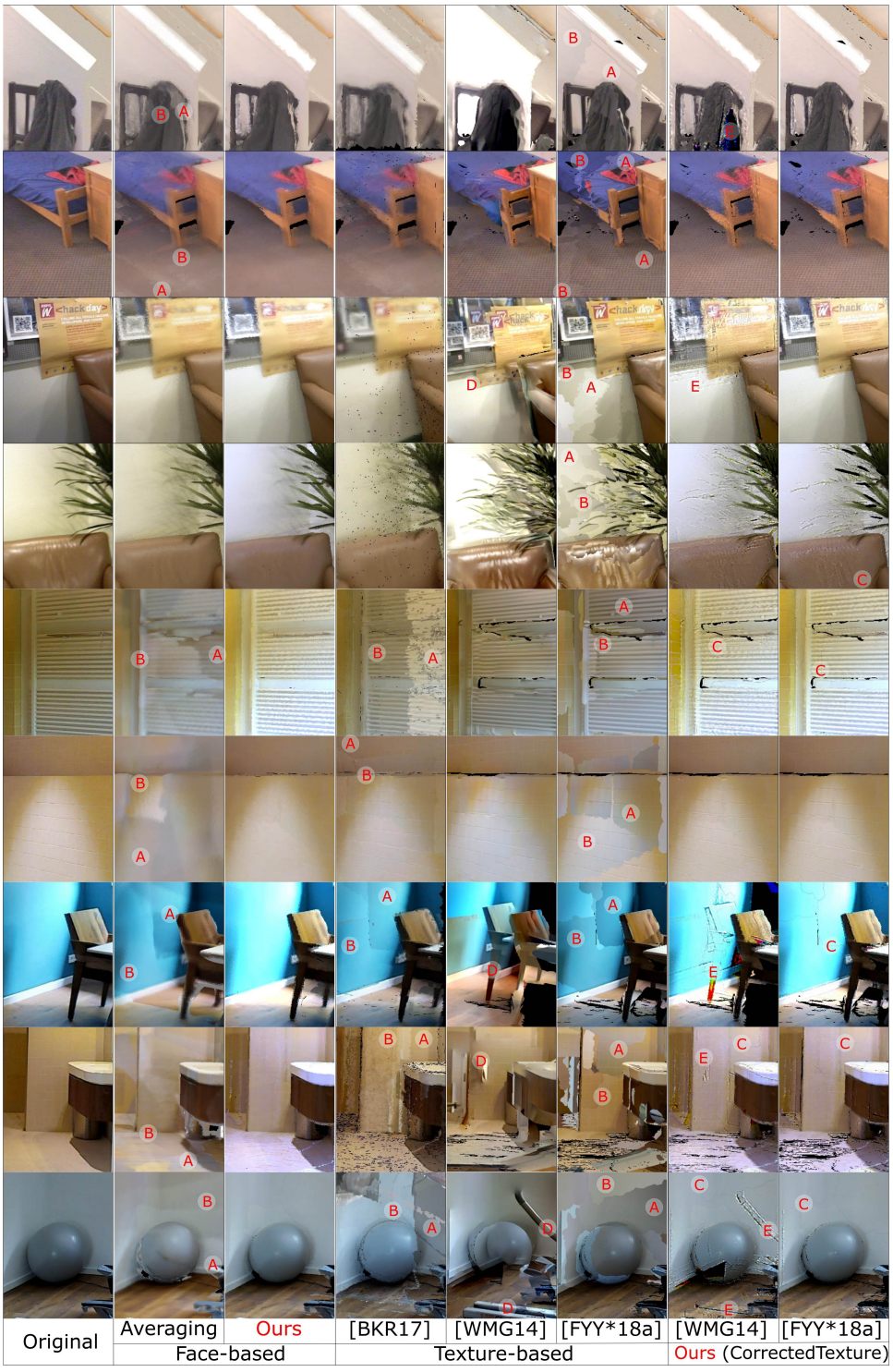


Figure 6.4: Comparison between our results as face-based or texture-based in (column 3, and (7,8) respectively) and other state-of-art techniques. First, two rows are results on a scene from [14], third and fourth rows are on scene from [15], while the rest are tests on different regions from scenes from Matterport3D [9] dataset.

6.3 Conclusions

We presented an efficient technique to reduce color discontinuities due to exposure and illumination differences in multi-view texturing by estimating consistent per-faces colors to serve as low-detailed colors for the mesh. Then we used it alongside texture patterns from previously generated textures from other techniques to provide consistent colors textures through a correction process, which offers flexibility to modify already textured models. We showed that our corrected texture quality is better than original textures from state-of-art texturing techniques even if there were inappropriate texture registrations.

Chapter 7

Future work:

Multi-View Light Source Localization

Light source localization using images is a difficult computer vision problem, specifically when light source is not visible on the images, yet important. Its difficulty derived from the possible lights, materials, shapes, and capturing device variations in the natural scenes. While knowing the location of the light source is considered one of the main cues that contribute to scene understanding. It allows for understanding how the shadows formed, how the colors grading, where specularities exist, and many other light dependent effects.

An image is the result of a complex interactions between the object shape, object material, light source, and capturing device. The light source is an important key for understanding surface properties e.g. surface structure and material property information. humans are using such information in their daily activities. For instance, a human can distinguish whether a surface is glossy/matte and what material the surface is made of e.g. metal or plastic or concrete, etc. Or can interpret the objects underlying geometry (due to shading cues). Shading indicates the curvature of the body and reveals the direction of the light. For forming and understanding images, a light source is an inevitable factor. Therefore, in this work, we focus on light source localization. Light Source Position (LSP) estimation or localizing

light sources (as a synonyms terminology) has caught a specific attention in works [131, 132, 133, 134, 135, 136, 137]. LSP estimation techniques assuming certain 3D-shapes of objects to estimate the position in the scene [8] (to obtain surface normals). Recently, using low cost RGB-D cameras sensors which acquire color images alongside with their depth alleviates the requirement of assuming certain object shapes, as the surface normals can be readily computed [138, 139, 140, 137, 141]. Assuming Lambert's law, the light source position could be estimated using pixel intensities and surface normals.

In particular, LSP is usually obtained by minimizing the residuals between the original scene and the re-rendered scene which is a scene generated using a hypothesized light source position. This is valid as long as Lambert's law holds, e.g matte surfaces. However, for a glossy surface that is prone to material-to-material inter-reflections and specular highlights which are not considered by Lambert's law, it will not be accurate. Moreover, due to the imperfections of capturing devices, the surface normals may be noisy on crinkled, rough, and grained surfaces. Hence, the LSP estimation may be negatively influenced by these types of surfaces.

In this work, giving a set of 2D images and its reconstructed 3D mesh either indoor or outdoor, our goal is to infer light sources (natural/artificial) positions through inverse rendering. As a first step toward solving the related problem, we infer its relative luminance through using our proposed mesh coloring technique in the previous chapter which overcomes the variations in illumination changing from multi-views caused by multiple scales of the same patches, moreover, estimating color from multi-view provide us with the advantage of overcoming the problems of inter-reflections and specular highlights. Thereafter using a new proposal for single texture mesh segmentation we divided the 3d mesh into segments each of them sup-

posed to have a single reflectance value for its surface material. After that, the relative approximation of scene illumination which is not affected by texture reflectances is estimated by balancing the luminance between mesh segments using the mesh balancing technique which overcomes the luminance variations in neighbor segments caused by different surface reflectances. Finally, we use the inverse radiosity equation for translating the relative illumination distribution over the patches to light source positions represented in terms of mesh patches.

7.1 Light Source Dataset

In this part of our work, we required a real-world annotated dataset to work on, to estimate and evaluate our LSP estimation. Light sources vary in its intensity and its way and angles of distribution which could be represented as Light distribution curve (LDC), so we needed to classify light sources into four classes (sun, artificial, window, and specular) light sources, each has its properties. Due to the lack of available real-world datasets for such problems, we had to annotate a real-world dataset for the evaluation purpose. We propose the Matterport3D Light Sources dataset (M3DLS) for multi-view light sources position detection which is based on the Matterport3D dataset [9]. M3DLS was annotated to four classes of light sources (sun, artificial, window, and specular) light sources. It contains 9 regions collected from Matterport3D scenes region segmentation, they are composed of (approx. 560 frames) annotated on the pixel level. See Figure 7.1 for a sample of annotations. The annotations are stored on a compatible convention with the Matterport3D dataset in "JSON" files.

Figure 7.1: Sample from the Matterport3D Light Sources dataset while left column is representing the original images, and the right one is representing the annotations projected on the image, we can see that the yellow color is representing direct light from the sun, while green is representing the artificial lights, cyan is for windows (opened which allow light to come in), and red colors that represent specular light types.

As we are working on 3D, we had to project our annotations from 2D images back to 3D mesh for evaluation, an example of our projected annotations to 3D-mesh is shown in Figure 7.2.

Figure 7.2: Sample from the M3DLS dataset, while the left column represents an original image from the scene showing some light sources, while in the right column our annotation (in yellow) projected on the 3D mesh (in blue).

7.2 Light source position estimation

We are trying to estimate the light source position (or more specifically faces of the mesh which are light sources) based on multi-view input data and its reconstructed mesh with its calibrated camera information. As shown in Figure 7.3, firstly, we want to clarify that there is a difference between falling light "illumination" and reflected light which could be represented as "Luminance" or "Value" channels in color models like International Commission on Illumination CIELAB, or (hue, saturation, value) Hsv, or (hue, saturation, intensity) Hsi, or others". The reflected

light is affected by scene elements reflectances that depend on material type, surface properties alongside the falling (affecting) illumination on the scene elements. We are concerned with the estimating an approximation of the falling illumination and use it to find the light source position.

Influenced by [142] our system is using the physical radiosity equation Eq. 7.2 in a reverse way to estimate the position of the light source instead of distributing from it by using their global model for radiosity using the linear system ($Fr = e$) as shown in Eq. 7.3. For being able to use this equation we should have a known scene reflectance ρ, illumination distribution over patches r, and relations between scene patches f. All of these unknowns will be estimated or approximated in our proposal.

In our proposal, we believe that the first step for accurate illumination approximation estimation, is to estimate consistent color distribution from multi-view as described in Chapter 6, Then from the estimated colors, we calculate the reflected luminance. Then influenced by Jorge et al. [115] we are trying to unify the reflectances ρ for the whole scene patches using our proposed single texture mesh segmentation as described in sec 7.2.1 by estimating albedo factor and use it to balance the luminance between segments. We consider this calculated (balanced) luminance as relative to the fallen illumination, and we can use it as its approximation in Eq. 7.3 as r vector to estimate the light source patches.

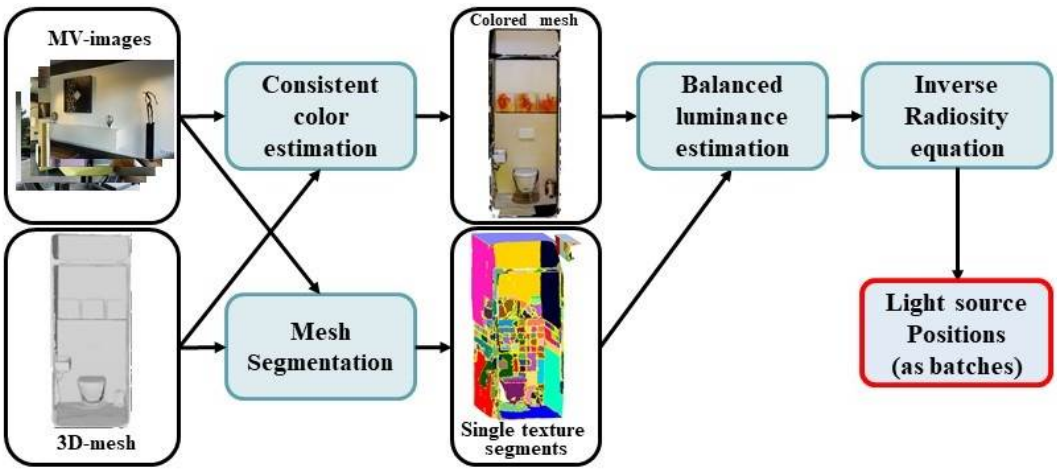


Figure 7.3: Proposed Light source position estimation system. At first we applied consistent color estimation technique and our (2D-3D) edge based segmentation technique to estimate a consistent colors for mesh and single texture segmented patches respectively. Later on we balance the estimated luminance from consistent mesh colors based on the single texture segmentations. Finally we use the balanced luminance in a reversed radiosity equation to estimate the light source position in terms of patches.

7.2.1 Mesh segmentation

After estimating consistent color distribution from multi-view images as described in Chapter 6 we need to segment the mesh into single texture segments so that we can be sure that each segment has a single reflectance value. For a single 2D frame we could segment it to several single texture segments using edge detectors, but to segment the 3D mesh to single texture segments we used the simple procedure shown in Figure. 7.4.

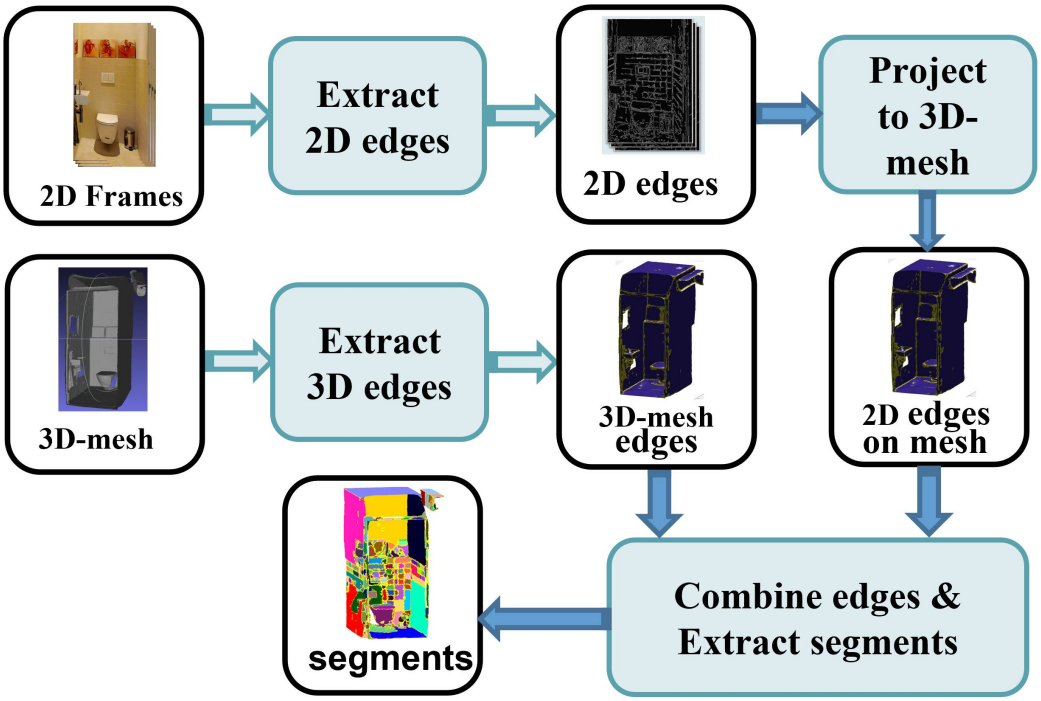


Figure 7.4: Proposed single texture segmentation procedure, we extract the edges from 2D images and 3D mesh and combine them together on the the 3D mesh to define the borders of our segments. while different colors in the segments mesh are representing different segment, which mean that all faces that share the same color are in the same segment.

We assume that changing of texture or surface reflectance can be detected in 2D image by the means of the high frequencies, which is represented as edges. So, for each single 2D frame we apply an edge detector on the luminance channel Y (from $Ycbcr$ color model) to detect the high frequencies in the image, we used the Roberts gradient operator to calculate the gradient magnitude. Then, using camera pose, intrinsic, and extrinsic parameters, we projected these edges to the 3D mesh and found its corresponding patches which will form the boundary of the mesh segments. 2D images borders may not fully cover all of the mesh patches which could yield some undesired connections between some different reflectance segments causing them to be segmented as one segment, so we also depended on the 3D mesh edges

alongside the 2D edges to segment our mesh. 3D edges are extracted from mesh based on the angle between patches. We search for the adjacent triangles which have a dihedral angle that deviates from π by an angle greater than filter angle α (flexible value-we choose based on the level of detection which we need). This method is typically used to extract the sharp edges in the surface mesh. Finally, we combine these two forms of edges in the 3D mesh as boundaries between segments. Finally, segments are considered as the connected faces groups which are not separated by boundaries.

7.2.2 Balancing estimated luminance

After estimating consistent mesh colors, the resulting color patches may be from different textures or different material which has different reflectance will be visible as strong color discontinuities. Thus, adjacent different reflectance segments need to be photometrically adjusted so that we are sure that the edges of adjacent segments have the same luminance to reduce light discontinuity. What is missing now is to estimate the reflectance for each segment to be able to estimate the approximation of the falling illumination, but without additional information, this would be difficult. So, we will try to solve it by unifying the reflectance of all segments so that we can consider the estimated luminance on them as an approximation for illumination, which we later can use in the global radiosity equation. In this section we are showing how we try to unify the reflectance of all segments by balancing the luminance between neighbor single reflectance segments. Simply we followed the work of Waechter et al [13] in their balancing between neighbor image segments to balance between our neighbour mesh segments. Models obtained from the mesh colors estimation contain many luminance discontinuities between segments due to the difference between their reflectances. These need to be adjusted to minimize

light discontinuity. We use the improved version of [13] for the global adjustment but applied it on patches instead of vertices. As shown in Figure 7.5, we believe that adjacent mesh patches should have nearby luminance as possible if their container segments have the same reflection, otherwise segments have different reflections.

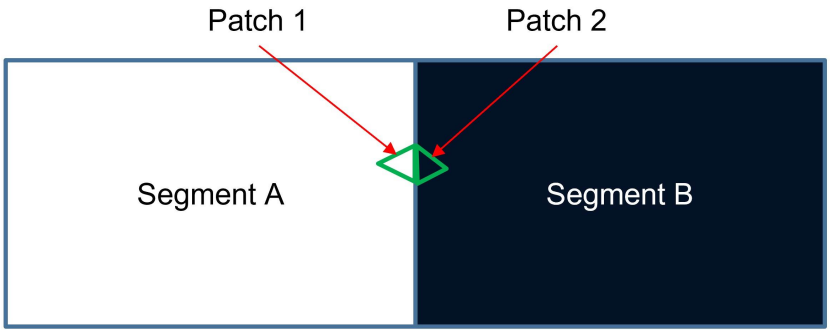


Figure 7.5: Neighbor patches should have same (or close) illumination values, otherwise segments have different reflections.

Based on this assumption, At first for each segment in the list of segments in our mesh $y = y_1, y_2, ..., y_u$, for each face (patch) in this segment from the set $F = f_1, f_2, ..., f_O$ which lie on the border of this segment, we try to find the nearest adjacent face from the adjacent segment to this face on the current segment. After this assignment, each face from a segment which lies on a border connected to another segment should have an assigned nearest adjacent face from another segment, see Figures [7.6, 7.7].

After assigning faces to their adjacent in other segments, we will organize the faces into two sets f_{left}, f_{right} which represent the set of faces and its correspondences in other segments respectively. to faces, we apply the optimization

$$\underset{g}{\operatorname{argmin}} \sum_f \left(f_{\text{left}} + g_{\text{left}} - (f_{\text{right}} + g_{\text{right}})\right)^2 + \frac{1}{\lambda} \sum_{b,j} \left(g_b - g_j\right)^2) \tag{7.1}$$

Where g is an additive value, and b, j are two corresponding faces. The first term ensures that the adjusted color to the left segment $(f_{left}+g_{left})$ and it's right adjacent $(f_{right} + g_{right})$ are as similar as possible while g is an additive value. The second term minimizes adjustment differences between adjacent faces within the same segment (texture patch). This favors adjustments that are as gradual as possible within a texture patch. After finding optimal g for all faces the corrections for each segment are interpolated from the g of its surrounding faces on its borders. Finally, the corrections are added to the final estimated mesh luminance.

7.2.3 Inverse Radiosity Equation

We have unified reflection segments alongside with approximated illumination distribution over the mesh, also, we will assume that all of the surfaces are diffuse. While the estimated balanced luminance is used as an approximation for the illumination, which we can use in the global radiosity equation Eq. 7.3 as the r vector to estimate the light source batches which is represented as e vector.

$$r_i = e_i + \rho_i \sum_{j=1}^{n-1} f_{ij} r_j, \tag{7.2}$$

where i and j index the n patches, e_i is a scalar for the self emittance of patch i ($e_i = 1$ for light source patches in the unit measure, 0 otherwise), ρ_i is the isotropic reflectivity of patch i and f_{ij} is the form factor between patch i and j.

$$\begin{bmatrix} 1-\rho_1 f_{11} & -\rho_1 f_{12} & \cdots & -\rho_1 f_{1n} \\ -\rho_2 f_{21} & 1-\rho_1 f_{22} & \cdots & -\rho_2 f_{2n} \\ \vdots & \vdots & \ddots & \\ -\rho_n f_{n1} & -\rho_n f_{n2} & \cdots & 1-\rho_1 f_{nn} \end{bmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix} \tag{7.3}$$

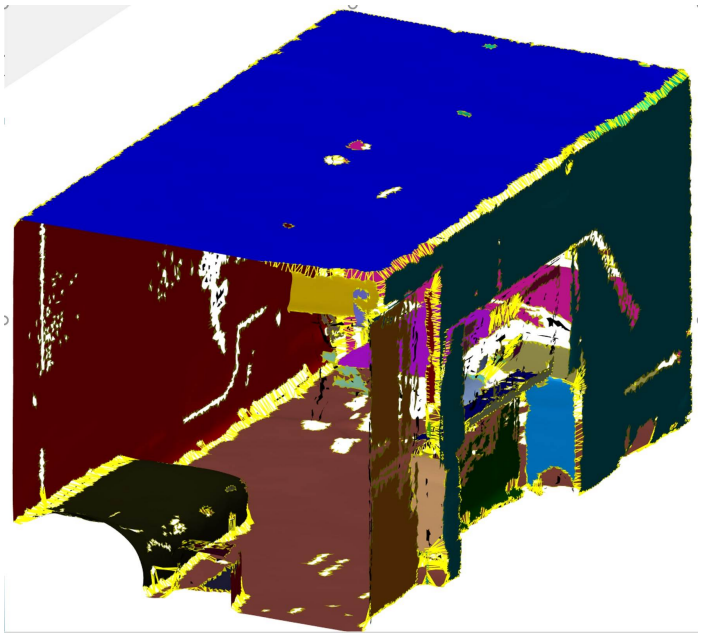


Figure 7.6: Different colors represent different segments, while yellow lines connect between faces and its adjacent faces in other segments.

7.3 Preliminary Experiments

In this section, we show our preliminary experiments results using our proposed system for light source position estimation. At first, in Figures [7.6, 7.7], We show some samples of mesh segmentation results, while each different color represent different group of faces which we refer to it as a segment, and we can see the yellow lines which connect the border faces in one segment to their correspondences border faces on the other segment.

Secondly, we show some samples of balancing mesh luminance in Figure 7.8. We can notice that in the first column which is the heat map of mesh luminance before balancing it is clear there are a clear difference between the luminance of the neighbour segments (see the second column to differentiate between segments), while in the third column we can notice that there are a smooth transition in the luminance

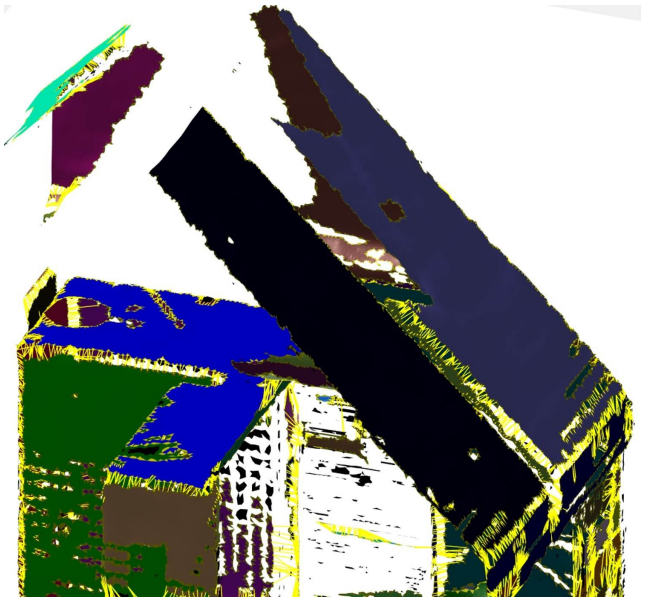


Figure 7.7: Different colors represent different segments, while yellow lines connect between faces and its adjacent faces in other segments.

between neighbour segments, which make the effect of light luminance more clearer (as full circles). These visual example give an indication that luminance balancing is going in the desired way.

At the end, in Figure 7.9, this heat-map for the illumination is estimated for a scene after applying light source position estimation steps in the proposed system presented in Figure 7.3.

We can notice that higher corners of the scene have higher values compared to others, even sometimes higher than the original light source patch. That happened because of missing regulations that regulate in which direction the lamp cast the light rays, which is called light distribution curve (LDC), moreover, more than one single area (we meant here the emitter area) can have direct contact with the same illuminated patches from the original light source which could result on a similar value like the original light source e.g. area beside Ground truth light source. More than that other areas can have direct contact with most illuminated patches in addition to the light

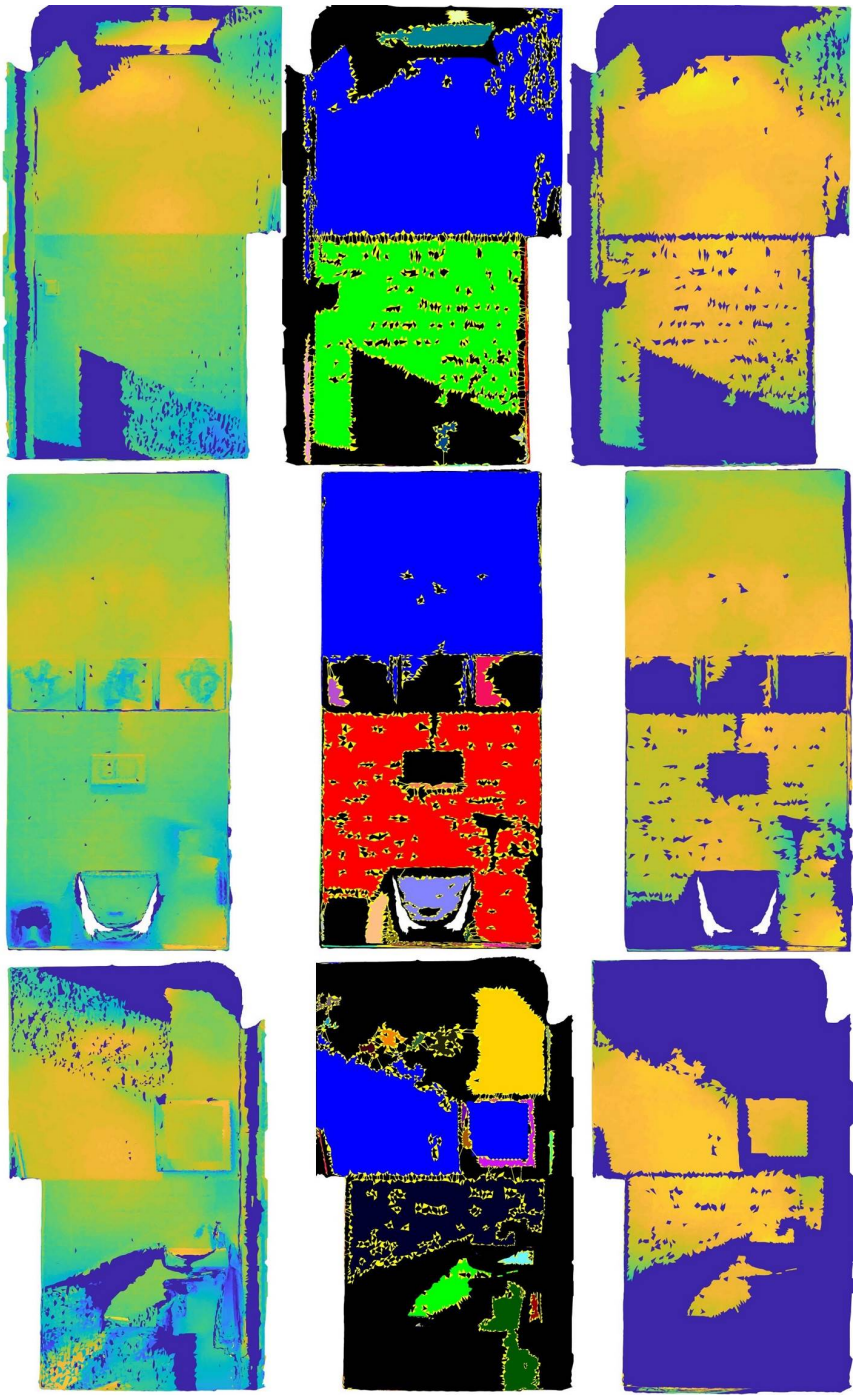


Figure 7.8: From left to right: heat map of mesh luminance before balancing, extracted single texture segments, and heat map of balanced luminance segments respectively. In the middle column, different colors represent different segments, while yellow lines connect between faces and its adjacent faces in other segments.

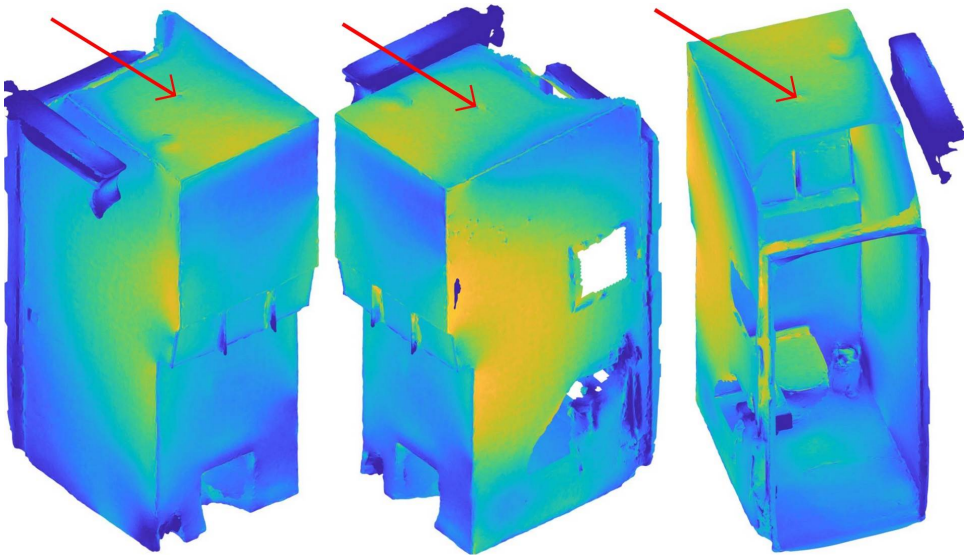


Figure 7.9: Result of estimating light source position, while the yellow color represents the intensity of the light, while the red arrows are pointing to the original light source position, we can see that the estimated light is concentrated on the sides or the corners of the scene.

source patch itself, which will yield values higher than the light source value itself.

7.4 Conclusions

For our future work, we are trying to estimate the light source position from the given information of a reconstructed multi-view scene. The preliminary results are showing some good results and others which are not encouraging, but we believe that this results can be enhanced by adding more regulations on how the light flows through the scene, either by extracting semantic information from multi-view images or by optimizing it in the 3D environment. While there is a lot of analysis that has to be done in our future work, especially that there are some areas should be revisited as it could be the reason for this inaccuracies as it has not been tested individually like segmentation stage, which we believe that it is also participating in this inaccuracies.

Chapter 8

Conclusion

In this thesis, we focus on providing techniques that leverage multi-view rich data to help computer vision techniques overcome visual errors effects towards better visual scene understanding. We showed that our proposed techniques provide better quantitative and qualitative results compared to state-of-art techniques. At first, we showed how the proposed aggregating methods for solving the multi-view color naming problem achieves high classification accuracy results. We showed that shadows affect the ability to classify color, and how excluding or correcting their corresponding affected pixels improves the classification accuracy from 75.45% to 79.09% on the M3DCN proposed dataset. Secondly, we targeted highlight specularities visual features, We have presented an effective and very fast technique (face-based) multi-view image specular detection algorithm which extends single image detectors. Using it we were able to distinguish between specular regions and non-specular regions which confuse the extended single image detectors. Despite that our accuracy is limited by face size and its count in the scene, but it was proven that it is still faster than and improved the accuracy by 3.5% over the baseline pixel-based techniques. Thirdly, We presented an efficient technique to reduce color discontinuities due to exposure and illumination differences in multi-view texturing by estimating consistent per-faces colors which we used later for correcting previously generated textures from other techniques to provide consistent colors textures which are vi-

sually pleasant and more realistic in compared to other techniques. Moreover, We also presented three detailed multi-view datasets (M3DCN, LIGHTs, and LSD) for evaluation purposes of these different multi-view problems.

In the future, our color-names estimation technique can be further extended to consider our detected specular highlights and analyze their performance. While our specularity detection technique can be applied to large-scale multi-view datasets (e.g. Matterport3D and Scannet) or incorporating semantic information to train deep models to overcome confusion between specularities and similar effects on single image problems.

Bibliography

[1] Y. Akashi and T. Okatani, "Separation of reflection components by sparse non-negative matrix factorization," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 611–625.

[2] J. Lin, M. E. A. Seddik, M. Tamaazousti, Y. Tamaazousti, and A. Bartoli, "Deep multi-class adversarial specularity removal," in *Scandinavian Conference on Image Analysis*. Springer, 2019, pp. 3–15.

[3] A. C. S. Souza, M. C. F. Macedo, V. P. Nascimento, and B. S. Oliveira, "Real-time high-quality specular highlight removal using efficient pixel clustering," in *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2018, pp. 56–63.

[4] H.-L. Shen and Z.-H. Zheng, "Real-time highlight removal using intensity ratio," *Appl. Opt.*, vol. 52, no. 19, pp. 4483–4493, Jul 2013. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-52-19-4483

[5] R. T. Tan and K. Ikeuchi, "Separating reflection components of textured surfaces using a single image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 178–193, 2005.

[6] Q. Yang, S. Wang, and N. Ahuja, "Real-time specular highlight removal using bilateral filtering," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg,

2010, pp. 87–100.

[7] T. Yamamoto and A. Nakazawa, "[papers] general improvement method of specular component separation using high-emphasis filter and similarity function," *ITE Transactions on Media Technology and Applications*, vol. 7, no. 2, pp. 92–102, 2019.

[8] D. Azinovic, T.-M. Li, A. Kaplanyan, and M. Nießner, "Inverse path tracing for joint material and lighting estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2447–2456.

[9] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3D: Learning from RGB-D data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[10] R. Guo, Q. Dai, and D. Hoiem, "Paired regions for shadow detection and removal," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2956–2967, Dec 2013.

[11] H. Le, T. F. Yago Vicente, V. Nguyen, M. Hoai, and D. Samaras, "A+d net: Training a shadow detector with adversarial shadow attenuation," in *European Conference on Computer Vision (ECCV)*, September 2018.

[12] S. Lin, Y. Li, S. B. Kang, X. Tong, and H.-Y. Shum, "Diffuse-specular separation and depth recovery from image sequences," in *Computer Vision — ECCV 2002*, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 210–224.

[13] M. Waechter, N. Moehrle, and M. Goesele, "Let there be color! large-scale texturing of 3d reconstructions," in *Computer Vision – ECCV 2014*, D. Fleet,

T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 836–850.

[14] Y. Fu, Q. Yan, L. Yang, J. Liao, and C. Xiao, "Texture mapping for 3d reconstruction with rgb-d sensor," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[15] Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 112:1–112:8, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2461919

[16] H. Rhodin, J. Spörri, I. Katircioglu, V. Constantin, F. Meyer, E. Müller, M. Salzmann, and P. Fua, "Learning monocular 3d human pose estimation from multi-view images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8437–8446.

[17] T. Isokane, F. Okura, A. Ide, Y. Matsushita, and Y. Yagi, "Probabilistic plant modeling via multi-view image-to-image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2906–2915.

[18] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki, "Gift: A real-time and scalable 3d shape search engine," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5023–5032.

[19] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3813–3822.

[20] T. Yu, J. Meng, and J. Yuan, "Multi-view harmonized bilinear network for

3d object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 186–194.

[21] A. Kanezaki, Y. Matsushita, and Y. Nishida, "Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5010–5019.

[22] T. Tlusty, T. Michaeli, T. Dekel, and L. Zelnik-Manor, "Modifying non-local variations across multiple views," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6276–6285.

[23] Y. Zhou and L. Shao, "Aware attentive multi-view inference for vehicle re-identification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6489–6498.

[24] S. Zhou, J. Wang, J. Wang, Y. Gong, and N. Zheng, "Point to set similarity based deep feature learning for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3741–3750.

[25] C. Su, J. Li, S. Zhang, J. Xing, W. Gao, and Q. Tian, "Pose-driven deep convolutional model for person re-identification," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3960–3969.

[26] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs, "Large scale multi-view stereopsis evaluation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 406–413.

[27] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang, "Deepmvs:

Learning multi-view stereopsis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2821–2830.

[28] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai, "Triplet-center loss for multi-view 3d object retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1945–1954.

[29] M. D. Elkhouly, S. James, and A. Del Bue, "Multi-view aggregation for color naming with shadow detection and removal," in *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*. IEEE, 2018, pp. 115–120.

[30] scratchapixel team. (2009) Learn computer graphics programming from scratch. [Online]. Available: https://www.scratchapixel.com/

[31] B. Berlin and P. Kay, *Basic color terms: Their universality and evolution*. Univ of California Press, 1991.

[32] J. M. G. Lammens, "A computational model of color perception and color naming," Ph.D. dissertation, State University of New York at Buffalo, Buffalo, NY, USA, 1995, uMI Order No. GAX95-09126.

[33] M. Seaborn, L. Hepplewhite, and J. Stonham, "Fuzzy colour category map for the measurement of colour similarity and dissimilarity," *Pattern Recognition*, vol. 38, no. 2, pp. 165 – 177, 2005.

[34] A. Mojsilovic, "A computational model for color naming and describing color composition of images," *IEEE Transactions on Image Processing*, vol. 14, no. 5, pp. 690–699, May 2005.

[35] E. van den Broek, T. Schouten, and P. Kisters, "Modeling human color cat-

egorization," *Pattern Recognition Letters*, vol. 29, no. 8, pp. 1136 – 1144, 2008, pattern Recognition in Interdisciplinary Perception and Intelligence.

[36] R. Benavente, M. Vanrell, and R. Baldrich, "Parametric fuzzy sets for automatic color naming," *Journal of the Optical Society of America A*, vol. 25, no. 10, pp. 2582–2593, Oct 2008.

[37] M. Serra, O. Penacchio, R. Benavente, and M. Vanrell, "Names and shades of color for intrinsic image estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 278–285.

[38] J. van de Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, July 2009.

[39] S. Liu, J. Feng, C. Domokos, H. Xu, J. Huang, Z. Hu, and S. Yan, "Fashion parsing with weak color-category labels," *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 253–265, Jan 2014.

[40] Y. Liu, Z. Yuan, B. Chen, J. Xue, and N. Zheng, "Illumination robust color naming via label propagation," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015, pp. 621–629.

[41] Z. Cheng, X. Li, and C. C. Loy, "Pedestrian color naming via convolutional neural network," in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 35–51.

[42] O. Russakovsky and L. Fei-Fei, "Attribute learning in large-scale datasets," in *European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes*, 2010.

[43] M. D. Elkhouly, S. James, and A. Del Bue, "Multi-view aggregation for color naming with shadow detection and removal," in *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, Dec 2018, pp. 115–120.

[44] G. D. Finlayson, M. S. Drew, and C. Lu, "Entropy minimization for shadow removal," *International Journal of Computer Vision*, vol. 85, no. 1, pp. 35–57, Oct 2009.

[45] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew, "On the removal of shadows from images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 59–68, Jan 2006.

[46] R. Guo, Q. Dai, and D. Hoiem, "Single-image shadow detection and removal using paired regions," in *CVPR*, June 2011, pp. 2033–2040.

[47] X. Huang, G. Hua, J. Tumblin, and L. Williams, "What characterizes a shadow boundary under the sun and sky?" in *International Conference on Computer Vision*, Nov 2011, pp. 898–905.

[48] T. F. Y. Vicente, M. Hoai, and D. Samaras, "Leave-one-out kernel optimization for shadow detection and removal," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 682–695, March 2018.

[49] T. F. Y. Vicente, M. Hoai, and D. Samaras, "Leave-one-out kernel optimization for shadow detection," in *IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 3388–3396.

[50] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, "Detecting ground shadows in outdoor consumer photographs," in *Computer Vision – ECCV 2010*,

K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 322–335.

[51] J. Zhu, K. G. G. Samuel, S. Z. Masood, and M. F. Tappen, "Learning to recognize shadows in monochromatic natural images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 223–230.

[52] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri, "Automatic feature learning for robust shadow detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1939–1946.

[53] L. Qu, J. Tian, S. He, Y. Tang, and R. W. H. Lau, "Deshadownet: A multi-context embedding deep network for shadow removal," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[54] X. Hu, L. Zhu, C.-W. Fu, J. Qin, and P.-A. Heng, "Direction-aware spatial context features for shadow detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[55] V. Nguyen, T. F. Y. Vicente, M. Zhao, M. Hoai, and D. Samaras, "Shadow detection with conditional generative adversarial networks," in *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 4520–4528.

[56] J. Wang, X. Li, and J. Yang, "Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[57] A. Artusi, F. Banterle, and D. Chetverikov, "A survey of specularity removal methods," *Computer Graphics Forum*, vol. 30, no. 8, pp. 2208–2230,

2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2011.01971.x

[58] G. J. Klinker, S. A. Shafer, and T. Kanade, "Using a color reflection model to separate highlights from object color," in *Proc. ICCV*, vol. 87. Citeseer, 1987, pp. 145–150.

[59] ——, "A physical approach to color image understanding," *International Journal of Computer Vision*, vol. 4, no. 1, pp. 7–38, 1990.

[60] ——, "The measurement of highlights in color images," *International Journal of Computer Vision*, vol. 2, no. 1, pp. 7–32, 1988.

[61] K. Schlüns and M. Teschner, "Fast separation of reflection components and its application in 3d shape recovery," in *Color and Imaging Conference*, vol. 1995, no. 1. Society for Imaging Science and Technology, 1995, pp. 48–51.

[62] K. Schluns and A. Koschan, "Global and local highlight analysis in color images," in *Proc. 1st Int. Conf. Color Graphics Image Processing*. Citeseer, 2000, pp. 300–304.

[63] R. Bajcsy, S. W. Lee, and A. Leonardis, "Detection of diffuse and specular interface reflections and inter-reflections by color image segmentation," *International Journal of Computer Vision*, vol. 17, no. 3, pp. 241–272, 1996.

[64] Y. Weiss, "Deriving intrinsic images from image sequences," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 68–75.

[65] T. Tan, K. Nishino, and K. Ikeuchi, "Illumination chromaticity estimation using inverse-intensity chromaticity space," in *2003 IEEE Computer Soci-*

ety Conference on Computer Vision and Pattern Recognition, 2003. Proceedings., vol. 1. IEEE, 2003, pp. I–I.

[66] R. T. Tan and K. Ikeuchi, "Estimating chromaticity of multicolored illuminations," in *Proceeding of the IEEE International Workshop on Color and Photometric Methods in Computer Vision*. Citeseer, 2003.

[67] H.-L. Shen and Q.-Y. Cai, "Simple and efficient method for specularity removal in an image," *Appl. Opt.*, vol. 48, no. 14, pp. 2711–2719, May 2009. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-48-14-2711

[68] S. W. Lee and R. Bajcsy, "Detection of specularity using color and multiple views," in *Computer Vision — ECCV'92*, G. Sandini, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 99–114.

[69] Tongbo Chen, M. Goesele, and H. . Seidel, "Mesostructure from specularity," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 1825–1832.

[70] S. K. Nayar, X.-S. Fang, and T. Boult, "Separation of reflection components using color and polarization," *International Journal of Computer Vision*, vol. 21, no. 3, pp. 163–186, 1997.

[71] D. W. Kim, S. Lin, K.-S. Hong, and H.-Y. Shum, "Variational specular separation using color and polarization." in *MVA*, 2002, pp. 176–179.

[72] R. T. Tan, K. Nishino, and K. Ikeuchi, "Separating reflection components based on chromaticity and noise analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1373–1379, 2004.

[73] W.-C. Ma, T. Hawkins, P. Peers, C.-F. Chabert, M. Weiss, and P. E. Debevec, "Rapid acquisition of specular and diffuse normal maps from polarized spher-

ical gradient illumination." *Rendering Techniques*, vol. 2007, no. 9, p. 10, 2007.

[74] F. Wang, S. Ainouz, C. Petitjean, and A. Bensrhair, "Specularity removal: A global energy minimization approach based on polarization imaging," *Computer Vision and Image Understanding*, vol. 158, pp. 31 – 39, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314217300474

[75] R. Feris, R. Raskar, Kar-Han Tan, and M. Turk, "Specular reflection reduction with multi-flash imaging," in *Proceedings. 17th Brazilian Symposium on Computer Graphics and Image Processing*, 2004, pp. 316–321.

[76] A. Agrawal, R. Raskar, S. K. Nayar, and Y. Li, "Removing photography artifacts using gradient projection and flash-exposure sampling," *ACM Trans. Graph.*, vol. 24, no. 3, p. 828–835, Jul. 2005. [Online]. Available: https://doi.org/10.1145/1073204.1073269

[77] B. Lamond, P. Peers, and P. E. Debevec, "Fast image-based separation of diffuse and specular reflections." *SIGGRAPH Sketches*, vol. 6, 2007.

[78] S. K. Nayar, G. Krishnan, M. D. Grossberg, and R. Raskar, "Fast separation of direct and global components of a scene using high frequency illumination," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 935–944.

[79] N. Kobayashi and T. Okabe, "Separating reflection components in images under multispectral and multidirectional light sources," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 3210–3215.

[80] K. Takechi and T. Okabe, "Diffuse-specular separation of multi-view images

under varying illumination," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 2632–2636.

[81] M. Haghighat, R. Mathew, and D. Taubman, "Rate-distortion driven decomposition of multiview imagery to diffuse and specular components," *IEEE Transactions on Image Processing*, vol. 29, pp. 5469–5480, 2020.

[82] M. Toscani, E. I. YÃ¼cel, and K. Doerschner, "Gloss and speed judgments yield different fine tuning of saccadic sampling in dynamic scenes," *i-Perception*, vol. 10, no. 6, p. 2041669519889070, 2019, pMID: 31897284. [Online]. Available: https://doi.org/10.1177/2041669519889070

[83] S. M. A. Shah, S. Marshall, and P. Murray, "Removal of specular reflections from image sequences using feature correspondences," *Machine Vision and Applications*, vol. 28, no. 3-4, pp. 409–420, 2017.

[84] D. N. Dövencioğlu, O. Ben-Shahar, P. Barla, and K. Doerschner, "Specular motion and 3D shape estimation," *Journal of Vision*, vol. 17, no. 6, pp. 3–3, 06 2017. [Online]. Available: https://doi.org/10.1167/17.6.3

[85] K. Doerschner, D. Kersten, and P. R. Schrater, "Rapid classification of specular and diffuse reflection from image velocities," *Pattern Recognition*, vol. 44, no. 9, pp. 1874 – 1884, 2011, computer Analysis of Images and Patterns. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320310004474

[86] Y. Adato, T. E. Zickler, and O. Ben-Shahar, "Toward robust estimation of specular flow." in *BMVC*, 2010, pp. 1–11.

[87] Shu-Kam Chow and Kwok-Leung Chan, "Specularity removal and relighting

of 3d object model for virtual exhibition," in *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[88] O. Yilmaz and K. Doerschner, "Detection and localization of specular surfaces using image motion cues," *Machine vision and applications*, vol. 25, no. 5, pp. 1333–1349, 2014.

[89] W. Li, S. Saeedi, J. McCormac, R. Clark, D. Tzoumanikas, Q. Ye, Y. Huang, R. Tang, and S. Leutenegger, "Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," *arXiv preprint arXiv:1809.00716*, 2018.

[90] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: 5m photorealistic images of synthetic indoor trajectories with ground truth," *arXiv preprint arXiv:1612.05079*, 2016.

[91] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, "A dataset and evaluation methodology for depth estimation on 4d light fields," in *Asian Conference on Computer Vision*. Springer, 2016, pp. 19–34.

[92] H.-L. Shen, H.-G. Zhang, S.-J. Shao, and J. H. Xin, "Chromaticity-based separation of reflection components in a single image," *Pattern Recognition*, vol. 41, no. 8, pp. 2461–2469, 2008.

[93] T. F. Y. Vicente, L. Hou, C.-P. Yu, M. Hoai, and D. Samaras, "Large-scale training of shadow detectors with noisily-annotated shadow examples," in *European Conference on Computer Vision*. Springer, 2016, pp. 816–832.

[94] M. Garon, K. Sunkavalli, S. Hadap, N. Carr, and J.-F. Lalonde, "Fast spatially-varying indoor lighting estimation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[95] M.-A. Gardner, K. Sunkavalli, E. Yumer, X. Shen, E. Gambaretto, C. Gagné,

and J.-F. Lalonde, “Learning to predict indoor illumination from a single image,” *arXiv preprint arXiv:1704.00090*, 2017.

[96] L. Murmann, M. Gharbi, M. Aittala, and F. Durand, “A dataset of multi-illumination images in the wild,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4080–4089.

[97] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, “Building rome on a cloudless day,” in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 368–381.

[98] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3d reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (ToG)*, vol. 32, no. 6, p. 169, 2013.

[99] Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. M. Seitz, “The visual turing test for scene reconstruction,” in *2013 International Conference on 3D Vision - 3DV 2013*, June 2013, pp. 25–32.

[100] Q.-Y. Zhou and V. Koltun, “Color map optimization for 3d reconstruction with consumer depth cameras,” *ACM Trans. Graph.*, vol. 33, no. 4, pp. 155:1–155:10, Jul. 2014.

[101] R. Gal, Y. Wexler, E. Ofek, H. Hoppe, and D. Cohen-Or, “Seamless montage for texturing models,” *Computer Graphics Forum*, vol. 29, no. 2, pp. 479–486, 2010.

[102] R. Pagés, D. Berjón, F. Morán, and N. García, “Seamless, static multi-

texturing of 3d meshes," *Comput. Graph. Forum*, vol. 34, no. 1, pp. 228–238, Feb. 2015.

[103] V. Lempitsky and D. Ivanov, "Seamless mosaicing of image-based texture maps," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–6.

[104] S. Bi, N. K. Kalantari, and R. Ramamoorthi, "Patch-based optimization for image-based texture mapping." *ACM Trans. Graph.*, vol. 36, no. 4, pp. 106–1, 2017.

[105] B. Wang, P. Pan, Q. Xiao, L. Luo, X. Ren, R. Jin, and X. Jin, "Seamless color mapping for 3d reconstruction with consumer-grade scanning devices," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 633–648.

[106] B. Burley and D. Lacewell, "Ptex: Per-face texture mapping for production rendering," *Computer Graphics Forum*, vol. 27, no. 4, pp. 1155–1164, 2008.

[107] C. Yuksel, J. Keyser, and D. H. House, "Mesh colors," *ACM Trans. Graph.*, vol. 29, no. 2, pp. 15:1–15:11, Apr. 2010.

[108] Z. Li, M. Shafiei, R. Ramamoorthi, K. Sunkavalli, and M. Chandraker, "Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image," *arXiv preprint arXiv:1905.02722*, 2019.

[109] H. Weber, D. Prévost, and J.-F. Lalonde, "Learning to estimate indoor lighting from 3d objects," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 199–207.

[110] A. Meka, M. Maximov, M. Zollhöfer, A. Chatterjee, H.-P. Seidel, C. Richardt,

and C. Theobalt, "Lime: Live intrinsic material estimation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[111] K. Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth, "Automatic scene inference for 3d object compositing," *ACM Trans. Graph.*, vol. 33, no. 3, pp. 32:1–32:15, Jun. 2014. [Online]. Available: http://doi.acm.org/10.1145/2602146

[112] B. J. Boom, S. Orts-Escolano, X. X. Ning, S. McDonagh, P. Sandilands, and R. B. Fisher, "Interactive light source position estimation for augmented reality with an rgb-d camera," *Computer Animation and Virtual Worlds*, vol. 28, no. 1, p. e1686, 2017.

[113] L. Zhang, Q. Yan, Z. Liu, H. Zou, and C. Xiao, "Illumination decomposition for photograph with multiple light sources," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4114–4127, Sep. 2017.

[114] E. Zhang, M. F. Cohen, and B. Curless, "Emptying, refurnishing, and relighting indoor spaces," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 174:1–174:14, Nov. 2016. [Online]. Available: http://doi.acm.org/10.1145/2980179.2982432

[115] J. Lopez-Moreno, E. Garces, S. Hadap, E. Reinhard, and D. Gutierrez, "Multiple light source estimation in a single image," in *Computer Graphics Forum*, vol. 32. Wiley Online Library, 2013, pp. 170–182.

[116] Y. Zhang and Y. . Yang, "Multiple illuminant direction detection with application to image synthesis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 915–920, Aug 2001.

[117] Y. Wang and D. Samaras, "Estimation of multiple illuminants from a sin-

gle image of arbitrary known geometry," in *Computer Vision — ECCV 2002*, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 272–288.

[118] C. . Bouganis and M. Brookes, "Multiple light source detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 4, pp. 509–514, April 2004.

[119] C. H. Kuo, S. Khamis, and V. Shet, "Person re-identification using semantic color names and rankboost," in *IEEE Workshop on Applications of Computer Vision (WACV)*, Jan 2013, pp. 281–287.

[120] L. Magerand and A. D. Bue, "Revisiting projective structure for motion: A robust and efficient incremental solution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018.

[121] J. J. Park, A. Holynski, and S. M. Seitz, "Seeing the world in a bag of chips," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[122] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[123] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[124] T. Yu, N. Xu, and N. Ahuja, "Shape and view independent reflectance map from multiple views," *International journal of computer vision*, vol. 73, no. 2, pp. 123–138, 2007.

[125] I. Nurutdinova, R. Hänsch, V. Mühler, S. Bourou, A. I. Papadaki, and O. Hellwich, "Specularity, shadow, and occlusion removal for planar objects in stereo case." in *VISIGRAPP (4: VISAPP)*, 2017, pp. 98–106.

[126] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1746–1754.

[127] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 12 Sept, 2017. [Online]. Available: https://www.blender.org/download/releases/2-79/

[128] Y. Fu, Q. Yan, L. Yang, J. Liao, and C. Xiao, "Texture Mapping for 3D Reconstruction with RGB-D Sensor," https://github.com/fdp0525/G2LTex, 2018.

[129] Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest (3d scenes)," http://qianyi.info/scenedata.html, 2013.

[130] J. Wu, H. Chen, X. Liu, L. Cao, X. Peng, and G. Jin, "Unsupervised texture reconstruction method using bidirectional similarity function for 3-d measurements," *Optics Communications*, vol. 439, pp. 85 – 93, 2019.

[131] Y. Wang and D. Samaras, "Estimation of multiple illuminants from a single image of arbitrary known geometry," in *European conference on computer vision*. Springer, 2002, pp. 272–288.

[132] A. O. Balan, M. J. Black, H. Haussecker, and L. Sigal, "Shining a light on human pose: On shadows, shading and the estimation of pose and shape," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[133] Q. Zheng and R. Chellappa, "Estimation of illuminant direction, albedo, and shape from shading," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 680–702, 1991.

[134] I. Sato, Y. Sato, and K. Ikeuchi, "Illumination from shadows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 3, pp. 290–300, 2003.

[135] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, "Estimating the natural illumination conditions from a single outdoor image," *International Journal of Computer Vision*, vol. 98, no. 2, pp. 123–145, 2012.

[136] C. B. Madsen and B. B. Lal, "Probeless illumination estimation for outdoor augmented reality," in *Augmented Reality*. Intech, 2010.

[137] L. Gruber, T. Richter-Trummer, and D. Schmalstieg, "Real-time photometric registration from arbitrary geometry," in *2012 IEEE international symposium on mixed and augmented reality (ISMAR)*. IEEE, 2012, pp. 119–128.

[138] B. Boom, S. Orts-Escolano, X. Ning, S. McDonagh, P. Sandilands, and R. B. Fisher, "Point light source estimation based on scenes recorded by a rgb-d camera." in *BMVC*, 2013.

[139] N. Neverova, D. Muselet, and A. Trémeau, "Lighting estimation in indoor environments from low-quality images," in *European conference on computer vision*. Springer, 2012, pp. 380–389.

[140] Z. S. Jiang, S. Rezvankhah, and K. Siddiqi, "Project report: Light source estimation using kinect," *Dept. Comput. Sci., McGill Univ*, 2013.

[141] L.-F. Yu, S.-K. Yeung, Y.-W. Tai, and S. Lin, "Shading-based shape refine-

ment of rgb-d images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1415–1422.

[142] T. Tsesmelis, I. Hasan, M. Cristani, F. Galasso, and A. Del Bue, "Rgbd2lux: Dense light intensity estimation with an rgbd sensor," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 501–510.

List of Publications Related to the Thesis

- **M. D. Elkhouly**, S. James, and A. Del Bue. Multi-view aggregation for color naming with shadow detection and re-moval. In 2018 IEEE International Conference on ImageProcessing, Applications and Systems (IPAS), pages 115–120, Dec 2018.

Awards Related to the Thesis

- Best paper award (IPAS conference - France 2018).