GEOMETRY-AIDED 3D IMAGE PROCESSING

By

Ze Zhang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science—Doctor of Philosophy

2023

ABSTRACT

The recent developments in 3D technology have profoundly impacted the digital world. Thus, the geometric modeling of 3D shapes through surfaces and curves plays an ever more critical role in society than before. Similarly, regularly sampled data on volumes (3D images) have found a wide range of applications in various areas, such as medicine, biology, engineering, military, entertainment, etc. Compared to 2D images, 3D images have one more dimension to store information. Therefore, they can directly approximate the physical world without first slicing up the object under investigation, which, however, leads to much more data complexity and technical difficulty. To address the problem of increased dimensionality, one may often leverage the lower dimensional geometric features abundant in various applications to facilitate computational tasks. Thus, how geometry could assist in 3D image processing is an intriguing topic, which we explore from three aspects in this dissertation.

With three real-world problems, we present novel geometry-aided 3D image processing algorithms with contributions in several areas, including biology, computer vision, and computer graphics. In these applications, we demonstrate how geometric structures are vital in guiding image processing, improving accuracy, and facilitating downstream analyses.

First, we analyze the 3D location-dependent fluorescence data of plant cells. We showed that with the volumetric diffusion using the 3D Laplacian matrix, we could produce an accurate adaptive local threshold to segment cytoskeleton in 3D microscope images of plant cells. Moreover, we propose several indices describing geometric and topological characteristics of cytoskeletons to help biologists understand actin filament dynamics in plant cells.

Second, we employ 3D-direction-dependent lighting conditions and introduce shadow masks to our face relighting pipeline generated from rough geometry to remove or add more accurate shadows for human face images. In addition, with the assistance of geometric information, we may generate relatively accurate spherical harmonics coefficients (a representation for low-frequency 3D-direction fields) to model the illumination for the relighting task.

Last, we explore how geometry shapes could help with a deep learning algorithm for a particular

field of both locations and directions, the dynamical neural radiance field for human heads. Given a set of input images or a video of a talking human head, we first embed a geometric model of the human head shape into the 3D implicit field. Then, one set of latent codes anchored on a morphable 3D surface mesh automatically turns the human face with any specific pose and expression into a radiance field. With the radiance field, it is straightforward to assemble RGB values for rays from arbitrary camera positions along chosen directions to form photorealistic rendering of novel views of the same person with changing expressions. Thus, we may reanimate the human head with better realism than a crude textured surface mesh. Furthermore, with the help of the radiance field, we may refine the geometric surfaces to align better with the input videos if surface meshes are needed for downstream applications.

Through the three examples of analyzing 3D spatial functions, 3D directional functions, and functions of both position and direction, we show the promises of geometry-assisted processing tools with concrete applications. Given the shape-dependent nature of most 3D images, we believe there are ample opportunities for integrating geometric representations into their processing pipelines.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

With the advances of 3D imaging technology and the availability of affordable 3D scanners, the amount of 3D data has increased tremendously. Compare to 2D images, 3D images provides us with rich information about the full geometry of 3D objects since it could best approximate the real-world objects without distortion. But the 3D image processing algorithms are usually expensive, computationally costly and math heavy. Data size will grow exponentially in 3D space with high resolution, therefore an efficient data structure and efficient processing algorithm are essential. The geometric and topological characteristic of 3D shapes are complex. Unlike 2D images, 3D data could have different representations, the geometric properties vary from one representation to another. How to analyse geometric features of the object, how to utilize the geometric information to render the shape into 2D image under different illumination, how to reconstruct the 3D mesh from 2D image, all of which are interesting topics with a lot of problems unsolved. In this document, we will explore what role geometry plays in solving the questions mentioned above.

In biology, scientists takes 3D microscopy images of plant cells and observe the structure of cytoskeleton to understand the dynamics of plant cells. Traditional ways of processing the 3D data is to project the 3D images to a 2D plane, which ignores the filament growing in vertical direction. To avoid information loss on Z axis, we propose a 3D algorithms to segregate cytoskeleton from the background. Due to the imaging mechanism of microscopy sensor, the intensity distribution of the filament is not globally consistent. We take the shape of filament into consideration, generate adaptive local thresholds by solving a partial differential equation to segment the cytoskeleton from the background as well as preserve all the geometric details on filament surface.

In computer vision, deep learning has gained notable success in 2D domain, but it is not fully employed on 3D data due to the complex nature of the 3D shapes. We also explore how the 3D geometric information could guide the learning of our network on face relighting task. It is a challenging problem to modify shadows in portrait images given a target lighting, since shadows are the result of interaction of 3D shapes and lighting condition in 3D space. With shadow maps

generated based on the face mesh under source lighting and target lighting, we could train our network to remove and add shadows in 2D portrait images.

View synthesis has been a long-standing and well-studied problem in Computer vision, with applications ranging from animation, image editing to VR, AR. Recent works mostly focus on learning the implicit representation for 3D objects. One big drawback is it requires dense input. We think it is possible to learn the representation for dynamic human head with sparse input leveraging the geometry of human head.

The rest of document is organized as follows. We first discuss the adaptive local thresholding algorithm and quantitative analysis on cytoskeleton in plant cells Chapter 2. Then, we demonstrate how shadow maps plays an important rule in face relighting task in Chapter 3. In Chapter 4, we shows how geometry helps with view synthesis, and we also discuss future work briefly.

CHAPTER 2

ILEE: AN ALGORITHM FOR QUANTITATIVE ANALYSIS OF CYTOSKELETAL IMAGES

2.1 Introduction

The eukaryotic cytoskeleton plays essential roles in cell signaling, trafficking, and motion in plant cells. Recent work towards defining the temporal and spatial dynamics of cytoskeletal organization, including as a function of cell status, has utilized quantitative analysis of cytoskeletal fluorescence images as a standard approach to defining cytoskeletal function. However, due to the uneven spatial distribution of the cytoskeleton, including varied filament shape and unstable binding efficiency to staining markers, these approaches may result in inaccurate cytoskeletal segmentation. Additionally, quantitative approaches currently suffer from human bias, as well as information loss caused by z-axis projection of raw images. To overcome these obstacles, we developed Implicit Laplacian of Enhanced Edge (ILEE), a cytoskeletal component segmentation algorithm, which uses a 2D/3D-compatible, unguided local thresholding approach, therefore providing less biased and more stable and accurate results. Empowered by ILEE, we constructed a Python library named ILEE_CSK, for automated quantitative analysis of cytoskeleton images, which computes cytoskeletal indices that cover density, bundling, severing, branching, and directionality. Compared to various classic approaches, the ILEE generates descriptive data with higher accuracy, stability, robustness, and efficiency. In addition to the analysis described herein, we have released ILEE_CSK as an open-source library for the community, together with Google Colab pipelines, as convenient and user-friendly access for biologists that requires no programming knowledge or specific computer configuration for usage.

2.1.1 Background

Higher eukaryotes have evolved complex mechanisms to organize and co-regulate a multitude of cellular processes, including growth, development, movement, cell division, and response to environmental stimuli. For example, plants coordinate growth with resistance against abiotic and biotic stress by engaging numerous systemic signaling processes, among which the cytoskeleton

plays an indispensable role [1]. To facilitate these processes and ensure robust and highly specific responses to changes in cell status, plants utilize two types of cytoskeleton – microfilaments and microtubules – to connect intercellular signaling to extracellular environments. Structurally, both are chains dynamically assembled from monomeric subunits named global actin and tubulin, respectively, and are involved in ceaseless events of polymerization/depolymerization, bundling, severing, and branching [2, 3], which is commonly referred to as "cytoskeletal dynamic". Spatially, the cytoskeleton forms a web-like matrix within the cytoplasm, and through its vast connectivity, functionally links the plasma membrane, numerous organelles, vesicles, and cellular environments – the sum of which serves as a cell surveillance and signaling platform that functions as a structural and information network [4]. As a structural component of the cell, the cytoskeleton controls numerous physical processes such as movement, shaping, cellular trafficking, and intercellular communication [5]. It also provides the mechanical force required for chromosome separation and plasma membrane division during mitosis and meiosis [6]. In addition to its role within the cytoplasm, the cytoskeleton is also required for a variety of functions within the nucleus, including RNA polymerase recruitment, transcription initiation, and chromosome scaffolding [7].

2.1.2 Related Work

Over the past several decades, confocal microscopy-based methods using fluorescence markers have been developed to monitor changes in cytoskeletal organization [8]. While showing advantages in real-time observation and intuitive visual presentation, these approaches possess critical limitations. They are subject to interpretation from captured images, which potentially involves human bias. As a step to remedy this limitation, the emergence of computational algorithm-based analyses offers a solution to describe the quantitative features of the cytoskeletal architecture with reduced human bias. However, while early studies introduced the concept of using generalizable image processing pipelines [9, 10] to transfer the task of evaluation away from the user and into a series of computer-based quantitative indices, several key bottlenecks emerged. First, most of the quantitative algorithms described to date are limited to 2D images. As a result, these approaches require the user to manually generate z-axis projections from raw data, resulting in an incredi-

ble amount of information loss, especially within the perpendicular portion of the cytoskeleton. Second, many approaches require users to manually set the threshold to segment cytoskeletal components from the images, resulting in sampling bias. Lastly, the accuracy and robustness of current algorithms greatly vary among different types of biological samples. This latter hurdle imposes a considerable disparity in the algorithm performance for plants (usually with curvy and spherical cytoskeleton) and animal (typically straight and complanate) samples, which compromises the performance of some advanced cytoskeleton analysis algorithms [11, 12, 13] when directly applied to plant cell images. In fact, while sample source dramatically impacts our ability to evaluate the features of cytoskeletal function across all eukaryotes, the vast majority of current approaches are developed based on cytoskeletal images from animal cells, which indicates potential systemic bias when applied to other types of image samples.

Previous work described the development of a global-thresholding-based pipeline to define and evaluate two key parameters of cytoskeleton filament organization in living plant cells: cytoskeletal density, defined by occupancy, and bundling, defined by statistical skewness of fluorescence [14]. Interestingly, while it utilizes manual global thresholding (MGT), which can potentially introduce a certain level of user bias, it still outperforms many standardized adaptive/automatic global or local thresholding approaches such as Otsu [15] or Niblack [16]. As a further advance of this early work, Higaki and colleagues developed the use of coefficient of variation (CV) of fluorescence to quantify the level of filament bundling, which improved the robustness and utility of the algorithm [17]. However, not only does this pipeline consume a considerable amount of time and effort from users for massive sample processing, but it also leaves unaddressed two key issues of rigor in image processing and analysis: information loss and human bias.

2.1.3 Our contribution

In the current study, we developed implicit Laplacian of enhanced edge (ILEE), a 2D/3D compatible unguided local thresholding algorithm for cytoskeletal segmentation and analysis, which is based on the native value, first-order derivative (i.e., gradient), and second-order derivative (i.e., Laplacian) of the cytoskeleton image altogether (see Fig. 2.1). The research described herein

supports ILEE as a superior quantitative imaging platform, one that overcomes current limitations related to information loss through dimensional reduction, human bias, and inter-sample instability. As shown, ILEE can accurately process cytoskeleton samples with a high dynamic range of fluorescence brightness and thickness, such as live plant samples.

As a key advance in the development of ILEE, we further established an ILEE-based Python library for the fully-automated quantitative analysis of 14 cytoskeletal indices within 5 primary classes: density, bundling, connectivity, branching, and anisotropy. This platform not only enables the acquisition and evaluation of key actin filament parameters with high accuracy from both projected 2D and native 3D images, but also improves the accessibility to a broader range of biologically-relevant states, including polymerization/depolymerization, bundling, severing, branching, and directional regulation. Herein, we introduce the core ILEE algorithm and propose several novel indices reflecting cytoskeletal dynamics. Using a defined series of images from multiple biological replicates of pathogen-infected plant cells, we demonstrate the performance of this algorithm by multi-perspective comparative analysis. Further, we provide evidence that supports the further advancement of 3D-based cytoskeletal computational approaches – a significant enhancement over currently available 2D-based approaches. Our library, ILEE_CSK, is publicly released at GitHub `https://phylars.github.io/ILEE_CSK`. In addition, we developed ILEE Google Colab pipelines for data processing, visualization, and statistical analysis, which is a convenient and user-friendly interface that requires no programming experience or particular computational device.

2.2 Implicit Laplacian of Enhanced Edge

2.2.1 Pipeline

Raw images generated by laser scanning confocal microscopy are typically obtained through detecting in-focus photons by a sensor from each resolution unit on a given focal plane. Since the cytoskeleton is a 3D structure that permeates throughout the cell, current approaches to capture filament organization and architectural parameters rely on scanning of each plane along the z-axis, independently, at regular intervals within a given depth, and reconstruction into 3D images.

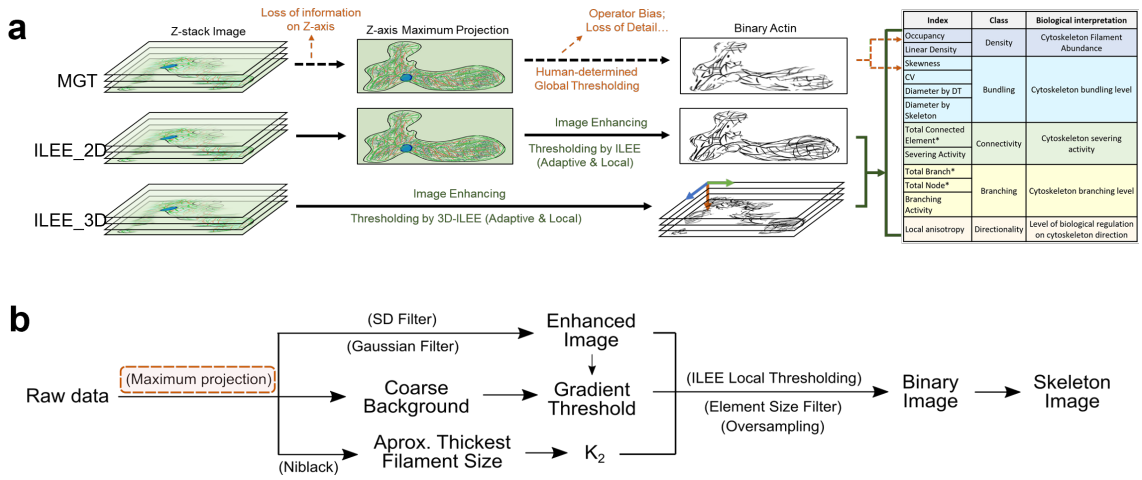


Figure 2.1 The pipeline of ILEE.

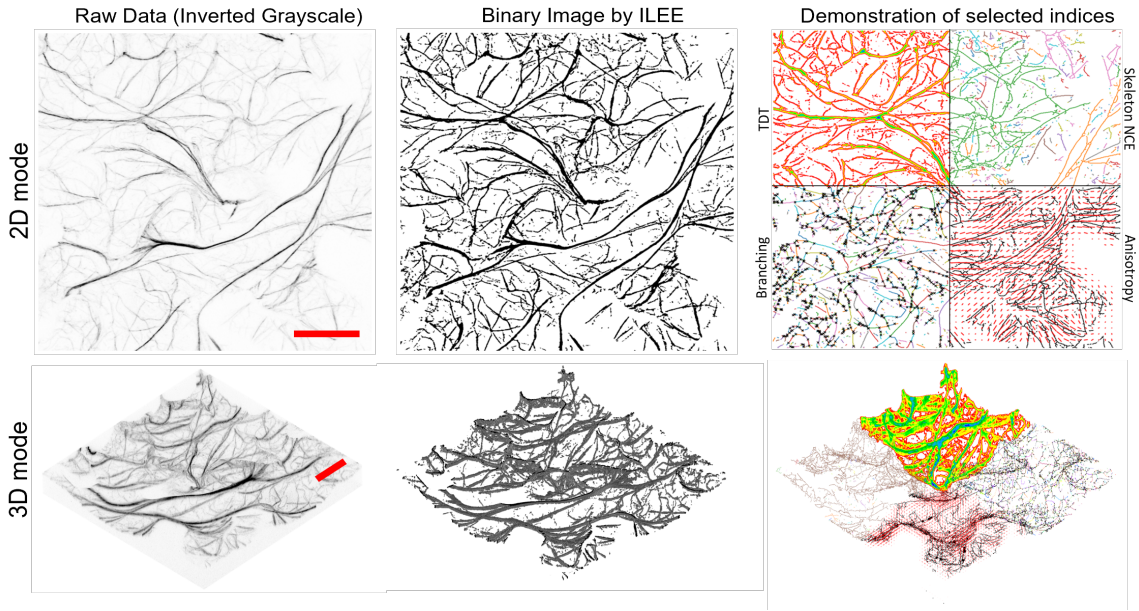
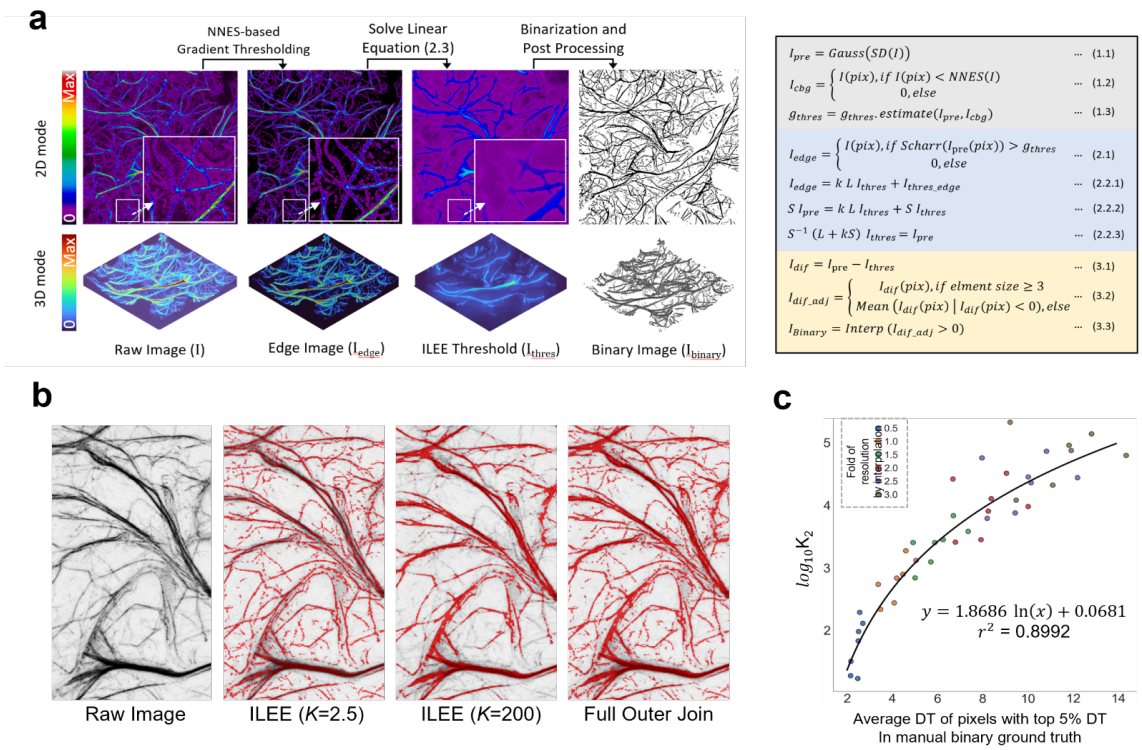


Figure 2.2 The input and output of ILEE.

However, due to limited computational biological resources, most studies have exclusively employed the z-axis projected 2D image, which results in substantial information loss, as well as systemic bias in downstream analyses.

In our newly developed algorithm, we integrated both 2D and 3D data structures into the same processing pipeline to ameliorate the aforementioned conflict (Fig. 2.1). In short, this pipeline enabled automatic processing and evaluation of both traditional 2D and native 3D z-stack image



$$I_{pre} = Gauss(SD(I)) \quad \cdots (1.1)$$

$$I_{cbg} = \begin{cases} I(pix), if\ I(pix) < NNES(I) \\ 0, else \end{cases} \quad \cdots (1.2)$$

$$g_{thres} = g_{thres}.estimate(I_{pre}, I_{cbg}) \quad \cdots (1.3)$$

$$I_{edge} = \begin{cases} I(pix), if\ Scharr(I_{pre}(pix)) > g_{thres} \\ 0, else \end{cases} \quad \cdots (2.1)$$

$$I_{edge} = k\ L\ I_{thres} + I_{thres\_edge} \quad \cdots (2.2.1)$$

$$S\ I_{pre} = k\ L\ I_{thres} + S\ I_{thres} \quad \cdots (2.2.2)$$

$$S^{-1}\ (L + kS)\ I_{thres} = I_{pre} \quad \cdots (2.2.3)$$

$$I_{dif} = I_{pre} - I_{thres} \quad \cdots (3.1)$$

$$I_{dif\_adj} = \begin{cases} I_{dif}(pix), if\ element\ size \geq 3 \\ Mean\ (I_{dif}(pix)\ |\ I_{dif}(pix) < 0), else \end{cases} \quad \cdots (3.2)$$

$$I_{Binary} = Interp\ (I_{dif\_adj} > 0) \quad \cdots (3.3)$$

Figure 2.3 Cytoskeleton segmentation by ILEE (a) steps of ILEE, (b) how coefficient k influences ILEE performance, (c) optimal K_2 estimation.

analysis. As shown in Fig. 2.1, cytoskeleton segmentation using ILEE requires 3 inputs: an edge-enhanced image, a global gradient threshold that recognizes the edges of potential cytoskeletal components, and the Laplacian smoothing coefficient K (described below). With these inputs, a local threshold image is generated via ILEE, and the pixels/voxels with values above the threshold image at the same coordinates are classified as cytoskeletal components. The output of this algorithm is a binary image (Fig. 2.2). Once acquired, the binary image is further skeletonized [18] to enable the downstream calculation of numerous cytoskeleton indices, the sum of which comprises the quantitative features of cytoskeletal dynamics (Fig. 2.2). Additionally, because the 2D and 3D modes share a common workflow, all of the calculated cytoskeleton indices also share the definition for both modes, regardless of the difference in dimensional spaces. This additional feature enables a horizontal comparison of both modes by the user, which we assert will significantly contribute to the community by providing massive image datasets for further examination, and comparison through the open-source library. In general, the ultimate goal of this approach, and resultant algorithm, is

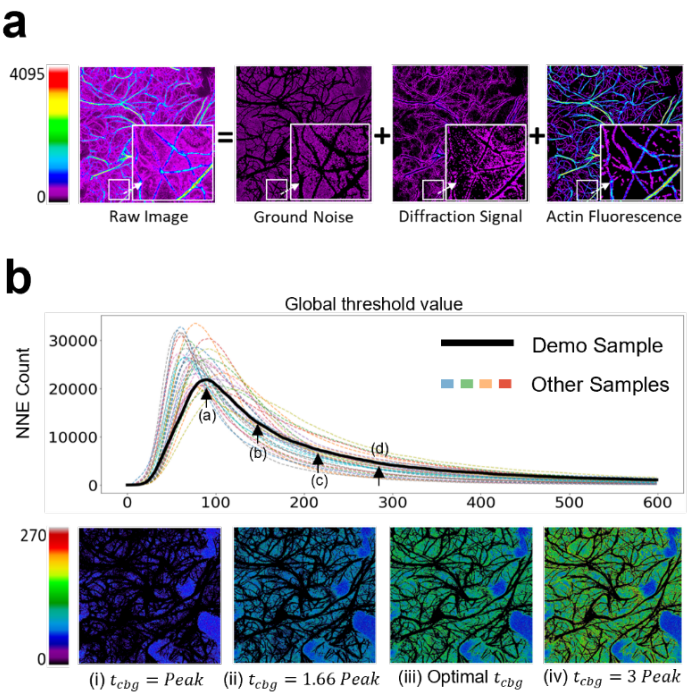


Figure 2.4 NNES global thresholding.

to construct a pipeline that enables the automated detection of the eukaryotic cytoskeleton from complex biological images in an accurate and unbiased manner.

2.2.2 ILEE

The pixel intensity of the cytoskeleton has a wide variation among samples, due to varied bundle thickness, the concentration of fluorescent dye, and its binding efficiency. Therefore, it is a highly challenging task to segment the cytoskeleton from the background. The pixel value generated by the confocal telescope is mainly influenced by three factors: 1. the real fluorescence emitted by the dye molecules. 2. the diffraction signal emitted by the actin filament in the neighboring space. 3. systematic error brought by the equipment. In our case, it is usually the ground noise generated by the imaging sensor. Since the photon sensor has a fixed setting, the distribution of ground noise is constant. But the diffraction signal varies when the thickness of the filament bundle in the local area changes. Therefore, an adaptive local thresholding algorithm is necessary to obtain reasonable cytoskeleton segregation results.

The key idea of ILEE is to solve a global partial differential equation (PDE) based on Laplacian

to generate a locally adaptive threshold for the cytoskeleton. One advantage of the Laplacian operator is that it could remove undesirable noise while still retaining desirable geometric features [19], which really helps tackle our problem. As we observe, the cytoskeleton usually has a very smooth surface and tubular shape. Therefore, ILEE could selectively filter out high-frequency noise while preserving salient geometric features of individual actin filaments, in leveraging the spectral characteristics of Laplace operators. We utilize the Laplacian function to build a global linear system. This way, the final result would be impacted by both the global shape and the local brightness level. Therefore we could avoid the drawback that local operators tend to restrict performance at varying filament thicknesses. Additionally, the edge of the cytoskeletal component is smoothed and elongated using a significant difference filter (SDF) and a Gaussian filter, the sum of which serves to enhance the continuity of the edge and contributes to the accuracy of edge detection (Fig. 2.1). Since we use an implicit method to solve the PDE function involving both the current state of the system and the later one, we name our algorithm Implicit Laplacian of Enhanced Edge (ILEE).

2.2.2.1 Linear System

We build the linear system based on a global Laplace operator. We add boundary constraints to our partial derivative equation to better preserve the geometric feature. First, we roughly estimate the edge pixel of actin filament based on gradient, because edges usually have very high gradient magnitude due to the dramatic change of the brightness. Then, we generate a selection set S with only points whose gradient is larger than the pre-estimated global threshold to mark the potential boundary:

$$S = \{(x, y, z)|\nabla|f(x, y, z)| > c\} \tag{2.1}$$

where c is the global threshold for edge pixels, $f(x, y, z)$ is the original image function.

Our goal is to preserve the edge points with high gradients, which serve as a guide for local thresholding:

$$g(x, y, z)|_s = f(x, y, z)|_s \tag{2.2}$$

After Laplacian processing, $g(x, y, z)$ should be harmonic. so we get:

$$\nabla^2 g(x, y, z) = 0 \tag{2.3}$$

Putting these two equations together to build our linear system. The final equation can be written as:

$$(L + kS)g = kSf \tag{2.4}$$

$$g = \frac{kS}{L + kS} f \tag{2.5}$$

where g is the input image, f denotes the output locally-adaptive threshold. L is the global Laplacian matrix, k is the coefficient that adjusts the influence the Laplacian has on the final result. S is the selection matrix, a diagonal matrix with the i-th diagonal entry being 1 if the i-th pixel has a norm of the gradient above the global gradient threshold.

We use the implicit Euler scheme to construct this differential equation to ensure the Laplacian operator behaves as a low-frequency filter to reduce high-frequency noise. The linear system could be solved efficiently since $A = L + kS$ is sparse. We use an off-the-shelf Matlab GPU linear solver (conjugate gradient). However, any generic linear solver can be used instead.

2.2.2.2 Global threshold

In order to identify ground noise and locate the background for downstream analyses (e.g., adaptive local thresholding), we designed an algorithm that calculates a global threshold using the morphological features of the ground noise, namely, non-connected negative element scanning (NNES). In brief, NNES calculates the total number of non-connected negative elements at different global thresholds, resulting in the identification of a representative value with a maximum non-connected negative element count (Fig. 2.4b (i)). The global threshold for the coarse background (Fig. 2.4b (iii)) will be determined using a linear model trained by the representative value rendered by NNES and manual global thresholding (MGT), a global threshold determined by operators experienced in cytoskeleton image analysis. NNES can maintain stability and accuracy over different samples that vary in the distribution of native pixel value because ground noise is the image component with the lowest value that is subject to a normal distribution and generally does not interfere with the actual fluorescence signal. Another accessible method is to directly use

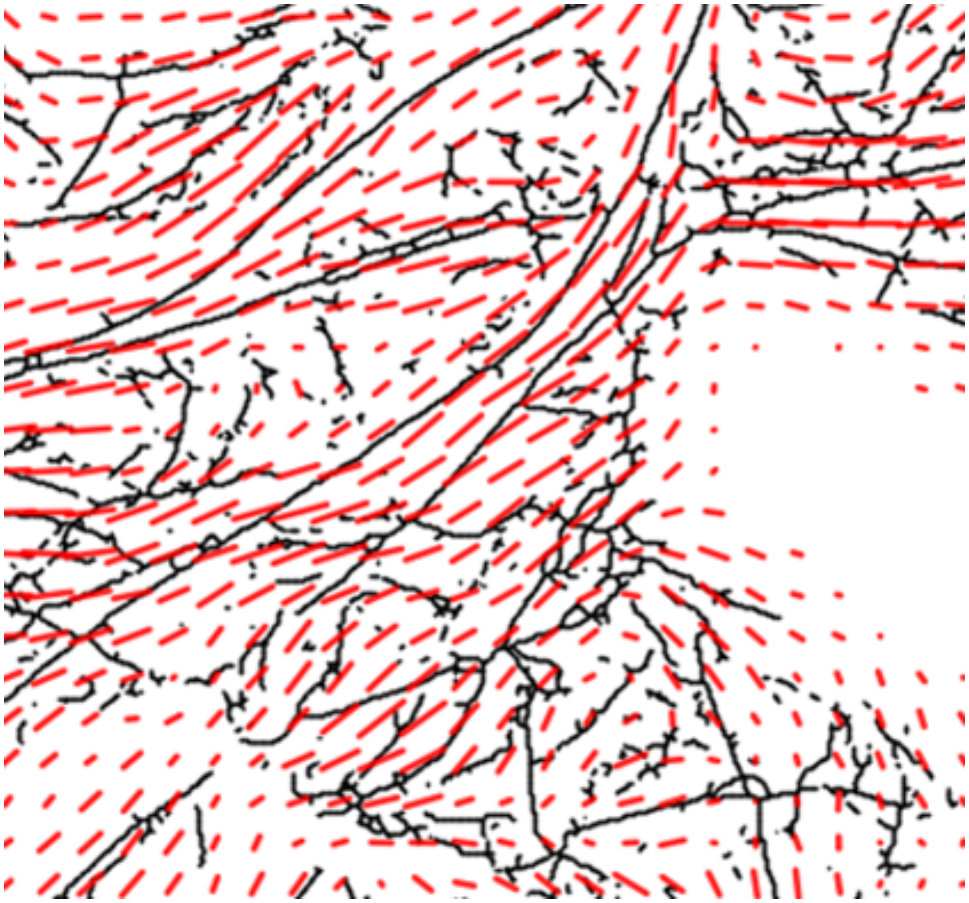


Figure 2.5 Local anisotropy.

the peak-of-frequency brightness of the image as a representative value to train a model. However, this approach is less accurate because the interval near the theoretical peak is always turbulent and non-monotone, a limitation potentially due to the pollution of diffracted light.

2.2.2.3 Global Laplacian matrix

Laplace operator is a second-order differential operator in n-dimensional Euclidean space. It is a measurement of how a point differs from its neighbor average. For each pixel, the Laplacian value is based on its immediate neighbors. In 3D space, it is defined as:

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = f_x + f_y + f_z \tag{2.6}$$

It is the sum of the second partial derivatives of the function with respect to each independent variable. In 3D Cartesian space, there are three independent variables: x, y, z. The derivatives on these three axes are defined as:

$$f_x = \frac{f(x+\Delta x, y, z) - 2f(x, y, z) + f(x-\Delta x, y, z)}{\Delta x^2} \tag{2.7}$$

$$f_y = \frac{f(x, y+\Delta y, z) - 2f(x, y, z) + f(x, y-\Delta y, z)}{\Delta y^2} \tag{2.8}$$

$$f_z = \frac{f(x, y, z+\Delta z) - 2f(x, y, z) + f(x, y, z-\Delta z)}{\Delta z^2} \tag{2.9}$$

If we discretize the continuous Laplacian function for 3D images, treat the whole 3D images as a discrete grid, Δx, Δy, and Δz are the sampling intervals along x, y, z axis, respectively. For our 3D image data, Δx and Δy would be pixel length and width, Δz is the gap between different image planes. To build the global Laplacian matrix, we transformed the 3D images into one n-dimensional vector v, and the corresponding Laplacian matrix is a large sparse $n \times n$ matrix written as (since the size of L matrix grows rapidly with 3D images, we take the L matrix for a 3×3 2D image for example):

$$L = \begin{bmatrix}
c & -\frac{2}{\Delta x^2} & & -\frac{2}{\Delta y^2} & & & & & \\
-\frac{1}{\Delta x^2} & c & -\frac{1}{\Delta x^2} & & -\frac{2}{\Delta y^2} & & & & \\
 & -\frac{2}{\Delta x^2} & c & & & -\frac{2}{\Delta y^2} & & & \\
-\frac{1}{\Delta y^2} & & & c & -\frac{2}{\Delta x^2} & & -\frac{1}{\Delta y^2} & & \\
 & -\frac{1}{\Delta y^2} & & -\frac{1}{\Delta x^2} & c & -\frac{1}{\Delta x^2} & & -\frac{1}{\Delta y^2} & \\
 & & -\frac{1}{\Delta y^2} & & -\frac{2}{\Delta x^2} & c & & & -\frac{1}{\Delta y^2} \\
 & & & -\frac{2}{\Delta y^2} & & & c & -\frac{2}{\Delta x^2} & \\
 & & & & -\frac{2}{\Delta y^2} & & -\frac{1}{\Delta x^2} & c & -\frac{1}{\Delta x^2} \\
 & & & & & -\frac{2}{\Delta y^2} & & -\frac{2}{\Delta x^2} & c
\end{bmatrix} \tag{2.10}$$

where $c = (\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} + \frac{2}{\Delta z^2})$. We use mirror technique to deal with boundary pixels.

2.2.2.4 Coefficient K

To determine the appropriate setting of K, we first tested how different K values influence the result of the local threshold image. As shown in Figure 2.3b, a low value of K generated a binary

image that is highly consistent with the selected edge. When K increases, the total threshold image shifted towards the average value of the selected edges with increasing loss of detail. A relatively lower K enables the accurate recognition of thin and faint actin filament components, yet is unable to cover the full width of thick filaments. Conversely, a high K value covers thick actin filaments with improved accuracy, resulting in a binary image that is less noisy; however, thin and/or faint filaments tend to be omitted as pseudo-negative pixels (Fig. 2.3b). To overcome this dilemma, we applied a strategy using a lower and a higher to compute two different threshold images, as well as binary images, that focuses on thin/faint components and thick components, respectively. Then, we generated a full outer-join image that contains all cytoskeleton components in these two binary images. This approach led to improved recognition of actin with varying morphologies (see Fig. 2.3b).

2.2.3 Analysis

Based on the binary image generated by ILEE, we developed an automatic tool to evaluate the quantitative features of the cytoskeleton. We first extracted the filament skeleton from the binary image, and then defined 12 indices to fully describe cytoskeleton from geometric, topological, and statistical aspects.

The skeleton, which is usually the medial axis of the actin filament, plays a vital role in cytoskeleton analysis. It indicates the growing direction and bundling density of the filament. It also reflects the topological structure of the cytoskeleton in a straightforward way. An accurate skeleton is also required to evaluate the filament thickness. To help with the downstream analysis, we applied the parallel thinning algorithm proposed by [18] to generate the skeleton for our sample data. The skeletonization algorithm takes a 3D binary image as input and builds an Euler table to remove surface points in all eight directions simultaneously. So we could obtain reasonable skeletons with desirable geometric and topological features preserved.

As the last step of our pipeline, cytoskeletal indices are automatically calculated from the binary image and skeleton generated by ILEE. As a substantial expansion from the previously defined cytoskeletal indices (e.g., occupancy, skewness, and CV), we propose 12 novel indices

within 5 classes. In short, these indices describe quantitative features of cytoskeletal morphology and dynamics, and each of these is critical considerations within the context of complex biological samples (Fig. 2.1a). It is worth noting that the accuracy of these indices could be enhanced with a certain level of image post-processing (e.g., oversampling)

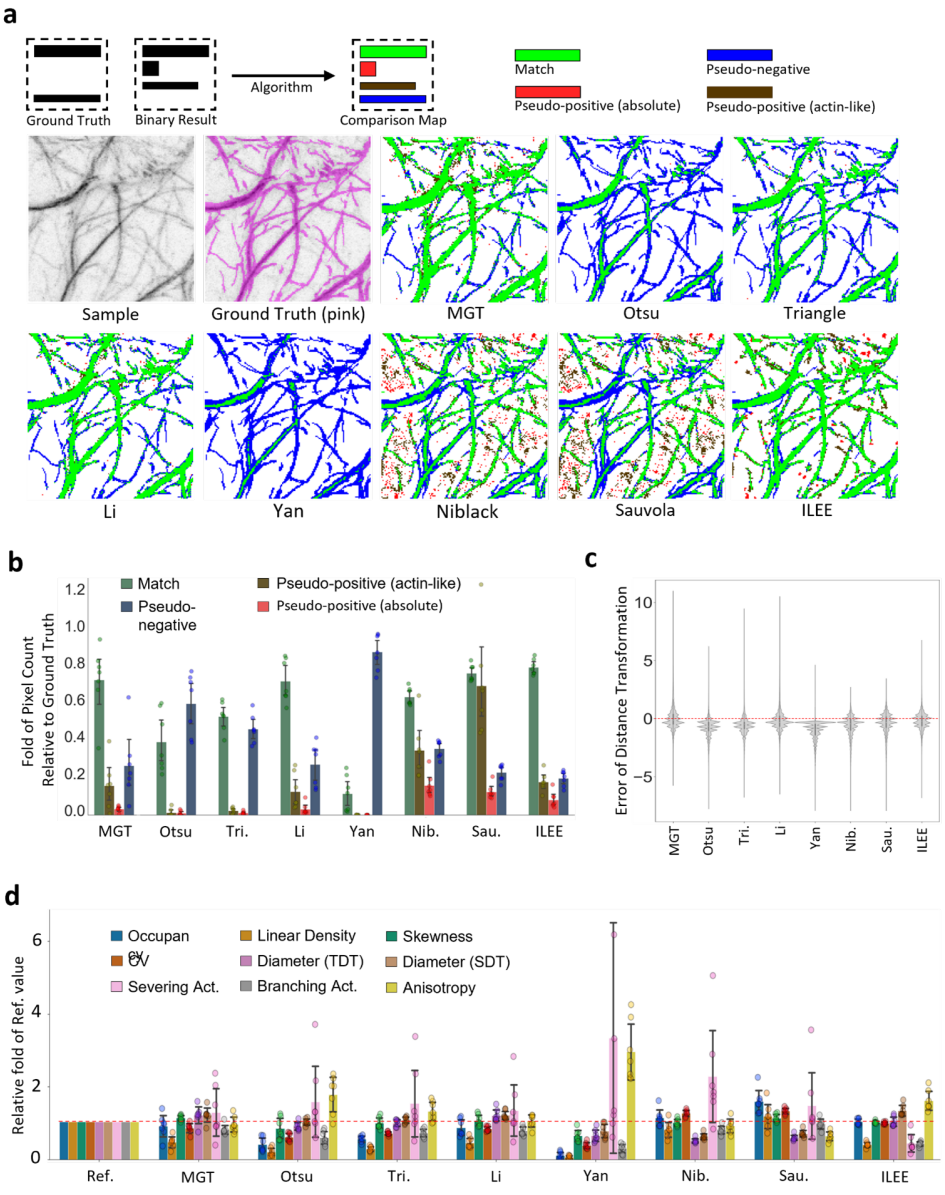


Figure 2.6 ILEE thresholding result.

Density: For the index class "density", we developed a novel set of metrics to evaluate linear density, a feature that measures filament length per unit of 2D/3D space.

• Occupancy: the frequency of the positive pixels in the binary image I_b generated by ILEE. We denote the number of pixels as N, the occupancy is written as:

$$O(I_b) = \frac{\sum I_b}{N} \tag{2.11}$$

• Linear Density: the length of filament per unit. Since cytoskeleton in plant cells are shaped like a cylinder, it is easier to estimate filament length on skeletonized image I_{sk}. We use the sum of the Euclidean lengths of all (graph theory defined) branches obtained by the Skan library [20] as the total length of the skeletonized filament and divide it by N.

Bundling: For "bundling", we developed two new, highly robust indices referred to as diameter-by-total DT (diameter TDT) and diameter-by-skeleton DT (diameter SDT), both of which measure the physical thickness of filament bundles, in addition to the indirect indices: skewness and CV, which estimate the relative bundling level based on the statistical distribution of fluorescence intensity.

• Skewness: the probability distribution of the filament pixel value with regarding its mean in the raw image I_r. Mathematically, it is defined as:

$$Skewness = \frac{1}{N_f} \sum_{I_b(x,y)==1} (\frac{I_r(x,y) - \mu}{\sigma})^3 \tag{2.12}$$

where N_f is the number of filament pixel numbers, μ is the mean, and σ is the standard deviation.

• CV: the coefficient of variance of filament pixels. It could be defined as:

$$CV = \frac{\sigma}{\mu} \tag{2.13}$$

• Diameter TDT: average filament diameter estimated by Euclidean Distance Transform of the whole binary image. The Euclidean distance transformation map I_dis stores the distance from one filament pixel to its nearest background pixel. For background pixel, its Euclidean distance would be 0.

• Diameter SDT: average filament diameter estimated by Euclidean Distance Transform on only filament pixels.

Connectivity: For the class "connectivity", we proposed two indices: total connected element and its derived index severing activity, which estimates the severing activities within per unit of the cytoskeleton. This additional metric assumes that severing generates two visible cytoskeletal filaments, which is distinguishable from filament depolymerization. This is an important consideration in terms of the biological activity of the cytoskeleton, as it enables the decoupling of the impact of filament depolymerization and filament severing, key activities of the eukaryotic actin-depolymerizing factor (ADF) and cofilin family of proteins [21].

• Severing activity: the number of connected components in binary image per unit length of the filament. So we get the number of connected components, then divide it by the total length of the filament skeleton to evaluate the severing activity.

Branching: For the class "branching", our algorithm is based on Skan, a recently developed Python library for the graph-theoretical analysis of the cytoskeleton [20]. To further explore the relationship between filament morphology and the biological activity of branching, we specifically designed an additional index, referred to as "branching activity", which we define as the total number of additional branches emerging from any non-end-point node per unit length of the filament. In total, this index measures the abundance/frequency of cytoskeletal branching.

• Branching activity: the number of branching point counts per unit length of filament skeleton. We first classify different types of branches. Only T type with three branches at one node and X type with four branches at one node is considered a non-end node. We only collect T-type nodes and X-type nodes and then divide the number by the total length of the filament skeleton.

Local anisotropy: Finally, our library is capable of estimating the level of directional cytoskeletal growth by indexing local anisotropy, which measures how the directionality of local filaments. We generate both numerical and visual output (Fig. 2.1c).

We performed a local averaging of filament alignment tensor, which is constructed as follows. First, we calculate the unit direction vector for each straight filament segment d_i. Then, the covariance matrix for each segment is obtained from the following equation:

$$t_i = d_i d_i^t \tag{2.14}$$

This rank-3 tensor is independent of the orientation of the filament segment, and can thus be averaged over a region containing a collection or unoriented line segments. We weigh each filament tensor in a circular/spherical local region by the length of every filament to produce a smoothed tensor field. The tensor field is written:

$$t = \frac{\sum_{i \in S} w_i t_i}{\sum w_i} \tag{2.15}$$

where t is the averaged tensor for the local region S. We calculate the eigenvectors and eigenvalues of t to describe the anisotropy for region S. The eigenvector corresponding to the largest eigenvalue indicated the primary orientation of filaments in this region as shown in Fig. 2.5. The difference between the maximum and the minimum eigenvalues is an indicator of the anisotropy in this region. If all the eigenvalues are the same, the indicator is 0, which implies an isotropic region with filaments growing towards random directions. If the eigenvalues other than the maximum are all nearly 0, all the filaments in this region are parallel to each other. In this case, they are all aligned with the maximum eigenvector, the dominant filament direction of this region.

2.3 Result

We designed a series of experiments to demonstrate the advantages of ILEE over traditional methods. We showed that ILEE could achieve high accuracy and stability over various samples.

ILEE achieves high accuracy over actin images. we constructed a dataset of actin images from Arabidopsis leaves with diverse morphology and compared ILEE with numerous traditional global and local thresholding algorithms. We also followed the traditional MGT pipeline asking some independent scientists with rich cytoskeleton analysis experience to manually set a global threshold for each sample and compared ILEE with the MGT result.

To evaluate the accuracy of each algorithm in terms of filament segregation, we manually generated the ground truth binary image from each sample, using a digital drawing monitor (Fig. 2.6a, ground truth). We used each of the ground truth binary images as a reference and compared the filament architecture obtained by ILEE, MGT, and 6 additional adaptive thresholding algorithms. These additional thresholding algorithms include Otsu[22], Triangle[23], Li[24], Yan[25], Niblack[16], and Sauvola[26] (Fig. 2.6). As an additional element of rigor, because

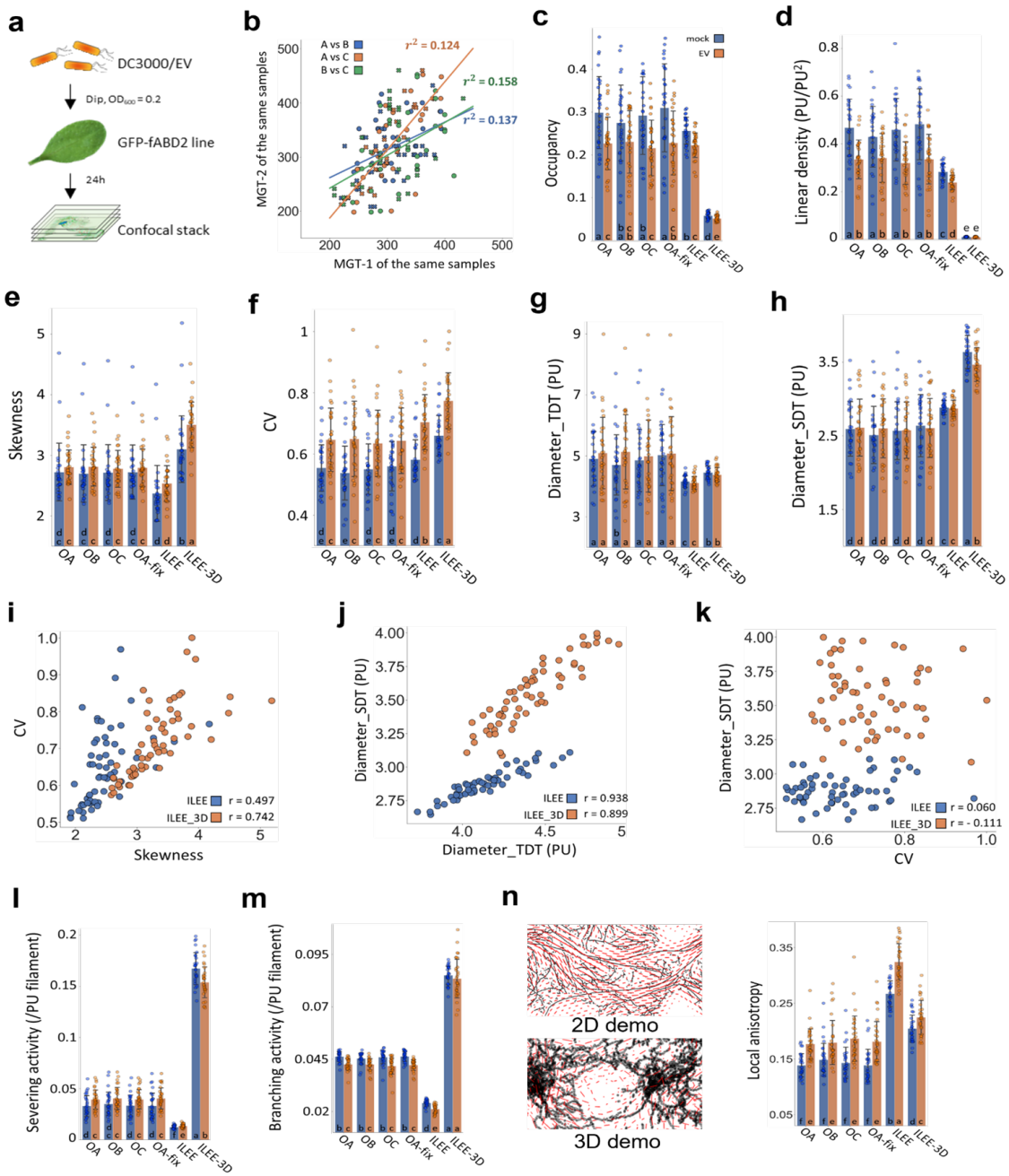


Figure 2.7 Quantitative analysis of cytoskeleton.

pseudo-positive pixels can be obtained due to user bias during the generation of the ground truth images (even when the operator is experienced in the actin imaging field), we further analyzed and categorized each non-connected component of pseudo-positive pixel by its shape and connectivity to matched elements, and identified the actin-like pseudo-positive pixels as possible real actin

components.

As shown in Fig. 2.6a (visualized demonstration), 2.6b (quantitative analysis), and 2.6c (bias analysis), ILEE offers superior performance, with the highest rate of accuracy with low pseudo-positive and pseudo-negative rates, as well as the lowest bias over local filament thickness. It is noteworthy, however, that the adaptive global thresholding approaches (from Otsu to Yan) tend to be relatively accurate when judging the thick and bright bundles of the cytoskeleton. However, these approaches are unable to capture faint filaments, and as a result, generate a high pseudo-negative rate. Conversely, both adaptive local thresholding approaches, Niblack and Sauvola, generate numerous block-shaped pseudo-positive elements and fail to capture the near-edge region of thick filaments. For MGT and Li method, although they showed satisfactory match rate, as well as lower, averaged pseudo-positive/negative rates, their performance is far less stable than ILEE (Fig. 2.6b).

As the next step in our analysis, we evaluated the accuracy and stability of cytoskeletal indices using ILEE versus other commonly used imaging algorithms. To do this, we first evaluated the ground truth indices from the manually generated binary images. In brief, quantitative measurements were collected from all methods and normalized by the relative fold to the result generated from the corresponding ground truth image. As shown in Fig. 2.6d, ILEE showed improved stability compared to all other quantitative approaches and the highest accuracy for occupancy, skewness, CV, and diameter.

However, we did observe that in terms of the morphology-sensitive indices (i.e., linear density, severing activity, and branching activity), the ILEE algorithm did not fully conform with data collected from the ground truth binary images. ILEE generated stable yet dramatically lower output compared to those derived from ground truth images (Fig. 2.6. Upon further inspection, we determined that this is because the manually portrayed ground truth images and ILEE results showed different tendencies in judging the pixels in the narrow areas between two bright filaments. Theoretically, human eyes could "hallucination" imaginary filaments that do not statistically exist. There are no criteria to assess which one is more accurate so far. Compare to other approaches, many displayed obvious, somewhat predictable, inaccuracies, the MGT and Li methods still gen-

erated satisfactory results, which echoes their performance in actin segmentation. However, the performance of these two algorithms among diverse and complex biological samples was not as stable as ILEE.

In order to further evaluate the stability and robustness of ILEE performance, we continued to analyze the variance coefficient of all groups (Supplemental Fig. 10), uncovering that ILEE is the only approach that simultaneously maintained high accuracy and stability. Next, we tested the robustness of ILEE and other approaches against noise signal disturbance by adding different levels of Gaussian noise to the image dataset (Supplemental Figs. 11-13). Using this approach, we observed that ILEE is still the best performing algorithm, maintaining stable and accurate results of image binarization and cytoskeleton indices against increasing noise. Taken together, these results demonstrate that ILEE has superior accuracy, stability, and robustness over MGT and other classic automated image thresholding approaches in terms of both cytoskeleton segmentation and index computation.

ILEE helps discover new features in response to bacterial infection. Our primary impetus for the creation of the ILEE algorithm was to develop a method to define cytoskeleton organization from complex samples, including those during key transitions in cellular status. For example, our previous research has demonstrated that the activation of immune signaling is associated with specific changes in cytoskeletal organization[27, 28]. Complementary to these studies, other research identified the temporal and spatial induction of changes in the cytoskeletal organization as a function of the pathogen (e.g., Pseudomonas syringae) infection and disease development[29, 30]. The sum of these studies, which broadly applied MGT-based quantitative analysis of cytoskeleton architecture, concluded that virulent bacterial infection triggers elevated density (by occupancy) yet induced no changes in filament bundling (by skewness) in the early stages of infection. Since one of our major motivations herein was to develop an ILEE-based toolkit supported by novel cytoskeletal indices to investigate the process of pathogen infection and immune signaling activation, we collected raw data from a previous study[31] describing a bacterial infection experiment using Arabidopsis expressing an actin fluorescence marker (i.e., GFP-fABD2), followed by confocal

imaging and data analysis by ILEE as well as MGT conducted by three independent operators with rich experience in actin quantificational analysis (Fig. 2.7). Additionally, because researchers sometimes apply a universal global threshold to all images from a batch of biological experiments to avoid tremendous labor consumption, we included this approach and aimed to observe its performance as well. In this experiment, the only categorical variant is whether sampled plants are treated with bacteria (EV) or not (mock). In total, nine indices that cover features of density, bundling, severing, branching, and directional order are measured and compared.

Our first concern is whether bias generated by MGT will influence the result and conclusion generalized from raw image samples of the experiment. We thereby analyzed the correlation of individual MGT values set by the three operators and found only a weak correlation between different operators (Fig. 2.7b), which indicates MGT bias indeed has the potential to impact quantitative results. Interestingly, while minor statistical discrepancies between MGTs by different operators are found in some indices (i.e., skewness and severing activity), most of the MGT results (both adaptive or fixed) shows the same trend as 2D ILEE, yet with far higher standard deviation, or lower stability (Supplemental Fig. 14a) over a certain biological treatment. This indicates that the historical data based on MGT should be majorly trustworthy despite the biased single data points, but an accurate conclusion must be based on a high sampling number that balances the deviation of individuals. Since ILEE provides more stable results over biological repeats, we are also interested in whether it renders higher statistical power to identify potential significant differences. Therefore, we compared the p-values of t-tests conducted for each index (Supplemental Fig. 14b) and found that ILEE indeed has the superior statistical power to distinguish numerical differences over datasets. We believe this demonstrates ILEE as a better choice for actin segregation.

Next, we attempted to understand whether different indices of the same class, particularly density and bundling, can reflect the quantitative level of the class in accordance, or instead show inconsistency. For density, we correlated the occupancy and linear density values of all methods over actin images of both mock and EV groups and found that occupancy and linear density measurements are in high conformity, with a Pearson coefficient at 0.98 (Supplemental Fig. 15).

Interestingly, while both demonstrate a high positive correlation, 2D ILEE and MGT do not share the same numeric relationship. Moreover, 3D ILEE has a weaker correlation, potentially due to cavities introduced by the skeleton image involved in linear density calculation. For bundling indices, we were interested in their level of conformity because direct indices (based on binary shape) and indirect indices (based on relative fluorescence intensity) are completely different strategies to measure bundling. Using the same approach of correlating analysis, we found that diameter_TDT and diameter_SDT indeed display strong positive correlation, while skewness and CV have merely medium-low correlation, which echoes the previous report demonstrating skewness and CV have different performance on the bundling evaluation. Unexpectedly, we also found that CV (as a representative of indirect indices) and diameter-SDT (as a representative of direct indices) have a striking correlation of zero. This is perplexing, as it raises the question of whether skewness or CV should be regarded as an accurate measurement of bundling (see Discussion). This discrepancy is also reflected by the result of 3D ILEE, whose CV and diameter-SDT over mock versus EV reveals the converse results at the significant difference. In general, we believe the biological conclusion that DC3000 treatment renders increased actin bundling level should be reconsidered with further inspection.

Last but not least, we sought to learn if additional features of plant actin cytoskeletal dynamics in response to virulent bacterial infection can be identified by the newly introduced indices and enhanced performance of ILEE. As shown in Fig. 2.7d, we observed significantly increased severing activity, local anisotropy, and decreased branching activity triggered by EV compared to the mock. At a minimum, these discoveries potentially lead to new biological interpretations, and as a result, may contribute to the identification of additional immune-regulated processes as a function of actin dynamics. However, while most of the 2D approaches were consistent and in agreement with the other indices, the severing activity estimated by 3D ILEE indicates a significant, but opposite, conclusion. After diagnosing the difference of each 2D ILEE and 3D ILEE sample, we concluded this is potentially due to information loss and misinterpretation by z-axis projection in the 2D-based approach. Therefore, we do not recommend totally depending on the 2D model

for the analysis of filament severing at the current stage and wish to gather more insight from the community in the future.

ILEE is compatible with various sample types. Cytoskeleton imaging from live plant samples is arguably one of the most difficult types of images to evaluate due to the dynamic topology and uneven brightness of actin filaments. While we demonstrated that ILEE shows superior performance over plant actin samples, ILEE and the ILEE_CSK library are generally designed for non-specific confocal images of the cytoskeleton and are therefore applicable to other types of samples. To investigate the compatibility of ILEE to other types of image samples, we tested ILEE on both plant microtubules35 and animal cell actin images (Supplemental Fig. 16). Importantly, we found ILEE can process, with high fidelity and accuracy, both plant and animal cytoskeletal features. This is encouraging, as animal cells generally possess a high volume of straight actin filament bundles, and therefore Hough transform-based feature detection is commonly applied to facilitate and/or enhance the performance of cytoskeleton segregation accuracy. However, this approach has certain limitations; specifically, they neglect and/or miscalculate curvy cytoskeleton fractions11,12. With the advancement of ILEE, Hough transform will not be absolutely necessary, and the potential cytoskeleton indices that rigorously require Hough transform can still utilize ILEE as a provider of binary image input for more accurate results.

In addition, we found that images of a single animal cell sometimes contain "void background" areas that are truly blank without any cellular component. This is different from plant tissue whose total image field is sample area, which may negatively influence the accuracy of the computed indices. To solve this issue and further support the animal cell sample, we developed a single-cell mode in the ILEE_CSK, which identifies the effective cell area using the statistical features of the brightness histogram and hence secures accurate index output. While ILEE was already tested on both plant and animal image samples, we would like to encourage researchers in the community to report issues (`https://phylars.github.io/ILEE\_CSK/Help\%20needed/`) where ILEE cannot segregate cytoskeleton correctly to help us improve the compatibility of ILEE and the ILEE_CSK library.

2.4 Conclusion and Future work

While ILEE has already remedied many disadvantages of traditional methods such as MGT, we are still working to further advance the ILEE approaches presented herein. Our goal is to ultimately arrive at a method that not only improves upon our currently described actin segmentation algorithms but also integrates time and space to describe a 4D model of cytoskeleton dynamics, as well as general cellular processes tractable by microscopy. As such, we offer the following as an initial list of potential upgrades and applications to be integrated into our library:

ILEE compatibility to x-y-t and x-y-z-t data, where t represents time. We are in the process of developing a 4D-compatible analysis of cytoskeletal dynamics that tracks filament organization over time. This approach will provide a temporal evaluation of supported indices with high accuracy and robustness.

Deep learning-based cytoskeleton segmentation algorithm with "foreign object" removal. As presented herein, ILEE enables the generation of trustworthy binary images on a large scale, which enables the construction of deep learning models to identify cytoskeleton components from confocal images with potentially better performance. The deep learning-based approach is also the key to solving the ultimate problem of all current cytoskeleton segmentation algorithms (including ILEE), which is the inability to detect and erase non-cytoskeleton objects with high fluorescence, such as the nucleus and cytoplasm. As one approach to circumvent this limitation

CHAPTER 3

SPHERICAL HARMONICS AND SHADOW MAPS FOR FACE RELIGHTING

3.1 Introduction

Face relighting is the problem of turning a source image of a human face into a new image of the same face under the desired illumination different from the original lighting. It has long been studied in computer vision and computer graphics and has a wide range of applications in face-related problems such as face recognition [32] as well as in entertainment. With the everlasting interest in consumer photography and photo editing, the ability to produce realistic relit face images will remain an important problem.

Many existing face relighting models utilize intrinsic decomposition of the image into face geometry, lighting, and reflectance [33, 34, 35, 36, 37, 32, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47]. The source image is then relit by rendering with a novel illumination. Other relighting methods employ image-to-image translation [48, 49, 34] or style transfer [50, 51, 52, 53].

For most face relighting applications, one important requirement is the preservation of the subject's local facial details during relighting. Intrinsic decomposition methods often compromise high-frequency details and can leave artifacts in the relit face images due to geometry or reflectance estimation errors. Another essential feature of a desirable face relighting model is proper shadow handling. For entertainment, in particular, adding and removing shadows accurately is crucial in producing photorealistic results. Most existing relighting methods, however, do not model *hard self-cast shadows* caused by directional lights.

Our proposed method uses an hourglass network to formulate the relighting problem as a ratio (quotient) image [54] estimation problem. In particular, the ratio image estimated by our model can be multiplied with the source image to generate the target image under the new illumination. Such an approach allows our relighting model to maintain the local facial details of the subject while adjusting the intensity of each pixel during relighting. Thus, we employ a ratio image estimation loss to enable ratio image learning, as well as a structural dissimilarity (DSSIM) loss based on the structural similarity metric (SSIM) [55] to enhance the perceptual quality of relit faces. In addition,

we incorporate PatchGAN [56] to further improve the plausibility.

During training, we generate and leverage shadow masks, which indicate estimated shadow regions for each image using the lighting direction and 3D shape from 3D Morphable Models (3DMM) [57] fitting. The shadow masks enable us to handle shadow through *weighted* ratio image estimation loss. We place a higher emphasis on the pixels close to shadow borders in the source and target relighting images, with larger weights placed on borders of high-contrast cast shadows over soft ones. This simple strategy allows learning how to accurately add and remove both hard and soft shadows under various relighting scenarios.

Our training process can leverage images with both diffuse and directional lighting across multiple datasets, which improves our ability to handle diverse lighting and generalize to unseen data over methods that only train on a single dataset [48, 34, 49]. To enable this, we use our shadow masks to estimate the ambient lighting intensity in each image and modify our lighting to account for differences in ambient lighting across images and datasets. Thus, our model accommodates differences in the environment between images in controlled and in-the-wild settings.

Our proposed method has three main contributions:

⋄ We propose a novel face relighting method that models both high-contrast cast shadows and soft shadows, while preserving the local facial details.

⋄ Our technical approach involves single image-based ratio image estimation to better preserve local details, shadow border reweighting to handle hard shadows, and ambient light compensation to account for dataset differences.

⋄ Our approach achieves the state-of-the-art relighting results on two benchmarks quantitatively and qualitatively.

3.2 Related Work

Face Relighting Among prior face relighting work, many conduct relighting via intrinsic decomposition and rendering [33, 34, 35, 36, 37, 32, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47]: the source image is decomposed into face geometry, reflectance, and lighting, and recombined with modified lighting to render relit images. As the decomposition generally relies heavily on single image

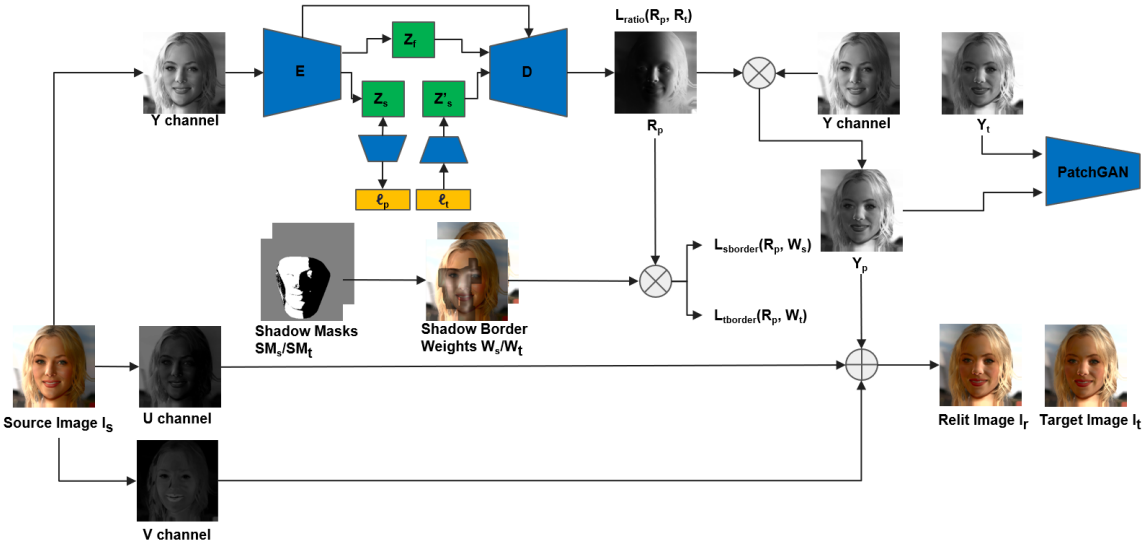


Figure 3.1 Overview of our proposed method.

face reconstruction, which remains largely an open problem, these methods tend to produce results that lack high-frequency detail found in the source image and contain artifacts from geometry and reflectance estimation error. Our method bypasses this issue by avoiding intrinsic decomposition entirely and estimating a ratio image instead. The ratio image only affects the intensity of each pixel in the source image in a smooth way aside from shadow borders, thus preserving the local facial details of the subject.

Other relighting methods do not perform explicit intrinsic decomposition. Sun *et al.* [48] use image-to-image translation to produce high-quality relighting. However, their results deteriorate when the input image contains hard shadows or sharp specularities, or when presented with strong directional light. Zhou *et al.* [49] assume a spherical harmonics (SH) lighting model and estimate the target luminance from the source luminance and a target lighting. However, their model is trained on images with primarily diffuse lighting and thus only handles soft shadows. We train our model on a mixture of images with diffuse and strong directional light. We also assign a larger emphasis on learning the ratio image near the shadow borders of high-contrast cast shadows. Hence, our model handles shadows effectively.

Some methods relight using a style transfer approach [50, 51, 52, 53] by transferring the lighting conditions of a reference image as a style to a source image. However, since they require a

reference image for each unique target lighting, they are less flexible to use in practice. Our model only requires a source image and a target lighting as input.

Ratio Images in Face Relighting Prior face relighting methods that incorporated ratio images often require multiple images per subject as input [54, 58] or both the source and target images [59], limiting their real-world applicability. Wen *et al.* [60] propose the first work on single image face relighting with ratio images, by estimating the ratio between the radiance environment maps. But they use a fixed face geometry with manually labeled feature correspondences while only considering diffuse reflections. We instead estimate the ratio image between the source and target images and thus can directly produce non-diffuse relighting without resorting to a face mesh. Zhou *et al.* [49] use a ratio image-based algorithm to synthesize their Deep Portrait Relighting (DPR) training data. Our work is the first ratio image-based face relighting method that can model non-diffuse relighting effects, including shadows caused by strong directional lights while requiring only one source image and a target lighting as input.

3.3 Method

3.3.1 Pipeline

Our model takes a source image and a target lighting as input and outputs the relit image under the target lighting along with the estimated source lighting. We represent the source and target lighting as the first 9 Spherical Harmonics (SH) coefficients. We adopt the hourglass network structure [49], but rather than directly estimating the target luminance, we instead estimate the ratio image between the input and target luminance. Similar to [49], our model only modifies the luminance channel: they use the *Lab* color space and estimate the target luminance before recombining with the source image's a and b channels to generate the relit image, whereas we estimate the ratio image for the Y channel of the *YUV* color space. The Y channel of the target image is then computed by multiplying the estimated ratio image with the Y channel of the source image, which is then recombined with the U and V channels of the source image and converted to *RGB* to produce the final relit image.

3.3.2 Training Losses

We employ several loss functions to estimate ratio images that preserve high-frequency details while capturing significant changes around shadow borders.

To directly supervise the ratio image learning, we employ the following ratio image estimation loss L_{ratio}:

$$L_{\text{ratio}} = \frac{1}{N} \| \log_{10}(\mathbf{R}_p) - \log_{10}(\mathbf{R}_t) \|_1. \tag{3.1}$$

Here, $\mathbf{R}_p$ and $\mathbf{R}_t$ are the predicted and ground truth ratio images, respectively, and N is the number of pixels in the image. Defining the loss in the log space ensures that ratio image values of r and $\frac{1}{r}$ receive equal weight in the loss.

We have two additional loss functions that place a higher emphasis on the ratio image estimation near the shadow borders of the source and target images. The shadow border ratio image loss $L_{\text{i,border}}$ is defined as:

$$L_{\text{i,border}} = \frac{1}{N_{\text{i}}} \| \mathbf{W}_i \odot (\log_{10}(\mathbf{R}_p) - \log_{10}(\mathbf{R}_t)) \|_1, \tag{3.2}$$

where i is s or t denoting the source or target respectively, and $\odot$ is element multiplication. $\mathbf{W}_s$ and $\mathbf{W}_t$ are per-pixel weights that are element-wise multiplied with the per-pixel ratio image error, enabling our model to emphasize the ratio image estimation at or near shadow borders. N_{s} and N_{t} are the number of pixels with nonzero weights in $\mathbf{W}_s$ and $\mathbf{W}_t$ respectively. See Sec. **??** for details on $\mathbf{W}_s$ and $\mathbf{W}_t$.

We also supervise the source lighting estimation using the loss term L_{lighting} defined as:

$$L_{\text{lighting}} = \| \ell_p - \ell_s \|_2, \tag{3.3}$$

where ℓ_p and ℓ_s are the predicted and ground truth source lighting, represented as the first 9 SH coefficients.

Similar to [49], we define a gradient consistency loss L_{gradient} to enforce that the image gradients of the predicted and target ratio images ($\mathbf{R}_p$ and $\mathbf{R}_t$) are similar, and a face feature consistency loss L_{face} to ensure that images of the same subject under different lighting conditions have the same

face features. L_{gradient} preserves the image edges and avoids producing blurry relit images. L_{face} further preserves the local facial details of the subject during relighting.

To enhance the perceptual quality of our relit images, we employ two PatchGAN [56] discriminators: one operates on 70×70 and the other on 140×140 patches. We train the discriminators jointly with the hourglass network using the predicted luminance as fake samples and the target luminance as real samples. We denote this loss as $L_{\text{adversarial}}$.

Finally, similar to [34], we define a structural dissimilarity (DSSIM) loss L_{DSSIM} between $\mathbf{R}_p$ and $\mathbf{R}_t$ as:

$$L_{\text{DSSIM}} = \frac{1 - \text{SSIM}(\mathbf{R}_p, \mathbf{R}_t)}{2}. \tag{3.4}$$

Our final loss function L is the sum:

$$L = L_{\text{ratio}} + L_{\text{sborder}} + L_{\text{tborder}} + L_{\text{lighting}} + L_{\text{gradient}} + L_{\text{face}} + L_{\text{adversarial}} + L_{\text{DSSIM}}. \tag{3.5}$$

We use coefficients to balance all the terms in the loss function.

3.3.3 Shadow maps

Spherical harmonics has long been used for diffuse and ambient lighting in Computer Graphics. SH lighting is a way to calculate the illumination on 3D models from image-based lighting sources in order to enable you to catch, relight and display global illumination style images. Spherical harmonics are camera-independent and require relatively low computational effort.

Spherical harmonics are the specific function defined on a unit sphere. They could be obtained by solving partial differential equations specifically Laplace's equation in the spherical domains. Since spherical harmonics form a complete set of orthogonal functions, any function defined on the surface of a sphere could be represented as the linear combination of spherical harmonics. All SH lighting techniques involve representing lighting equations with spherical functions that have been projected into frequency space using the spherical harmonics as a basis. Spherical harmonics have been commonly used in modeling illumination for Lambertian reflectance. Previous work [49] uses SFS network to estimate the SH coefficients. Since we are aware of either the light direction or the point light position for our datasets, we could calculate SH coefficient numerically to avoid

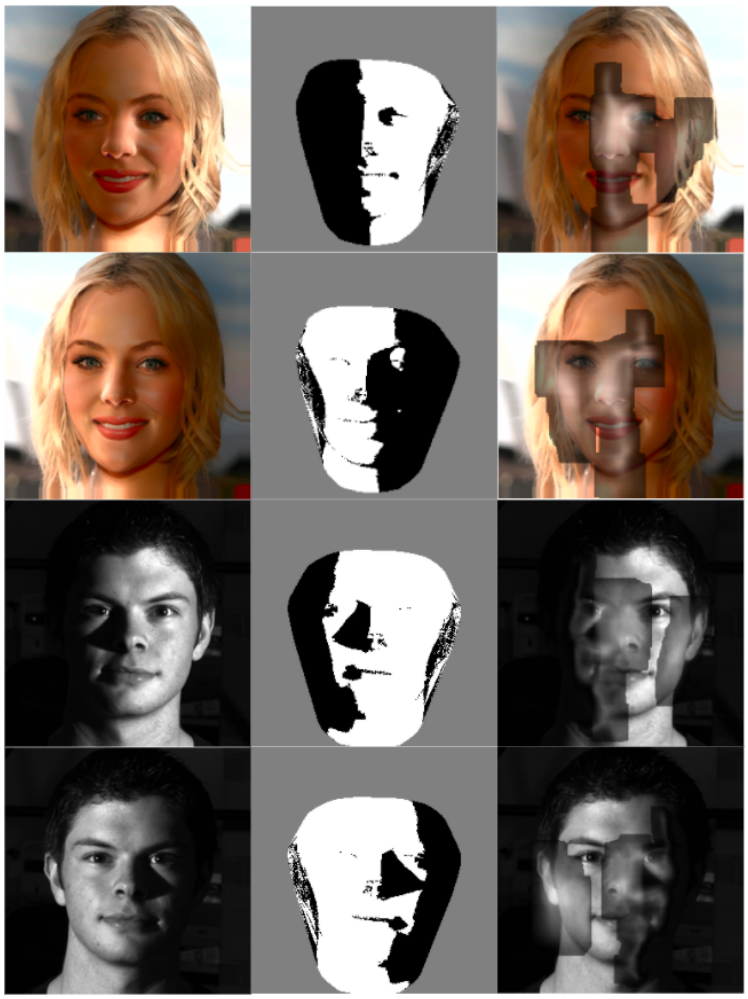


Figure 3.2 Shadow Masks and Weights. (a) input image, (b) shadow mask, (c) shadow border weights (intensity proportional to weights).

the estimation error produced by the neural network. We propose a method to estimate precise SH coefficients. We first create an environment map on a unit sphere according to the lighting condition, and then we project the environment map to SH basis to generate the SH coefficients for our dataset. The environment map is reduced to the first nine SH coefficients since high-frequency lighting details are not required in our case.

SH basis: The spherical harmonics $Y_l^m(\theta, \phi)$ are the angular portion of the solution to Laplace's equation in spherical coordinates. Spherical harmonics take their simplest form in the Cartesian coordinate system. They can be defined as homogeneous polynomials of degree l, order m at

(x, y, z) that satisfy Laplace's equation. The set of real spherical harmonics are written as:

$$
\begin{aligned}
Y_0^0 &= \frac{1}{2}\frac{1}{\sqrt{\pi}} \\
Y_1^{-1} &= \frac{1}{2}\sqrt{\frac{3}{\pi}}\frac{x}{\sqrt{x^2+y^2+z^2}} \\
Y_1^0 &= \frac{1}{2}\sqrt{\frac{3}{\pi}}\frac{z}{\sqrt{x^2+y^2+z^2}} \\
Y_1^1 &= \frac{1}{2}\sqrt{\frac{3}{\pi}}\frac{y}{\sqrt{x^2+y^2+z^2}} \\
Y_2^{-2} &= \frac{1}{2}\sqrt{\frac{15}{\pi}}\frac{xy}{x^2+y^2+z^2} \\
Y_2^{-1} &= \frac{1}{2}\sqrt{\frac{15}{\pi}}\frac{yz}{x^2+y^2+z^2} \\
Y_2^0 &= \frac{1}{4}\sqrt{\frac{5}{\pi}}\frac{-x^2-y^2+2z^2}{x^2+y^2+z^2} \\
Y_2^1 &= \frac{1}{2}\sqrt{\frac{15}{\pi}}\frac{zx}{x^2+y^2+z^2} \\
Y_2^2 &= \frac{1}{2}\sqrt{\frac{15}{\pi}}\frac{x^2-y^2}{x^2+y^2+z^2}
\end{aligned}
\tag{3.6}
$$

where (x,y,z) is the unit vector on the unit sphere.

Environment map: Similar to [61], we model our light as a Gaussian light (where mean position μ is the light position in camera coordinates and standard deviation $\sigma = 8°$). Then we project the light to a unit sphere centering at origin to generate a 256×128 environment map. The pixel intensity of the environment map is defined as:

$$
I(\theta, \phi) = \begin{cases} 0 & \theta > 90° \\ e^{-\frac{\alpha^2}{2\sigma^2}} & 0° \leq \alpha \leq 90° \end{cases}
\tag{3.7}
$$

where α is the angle between the light direction and the unit vector of inclination θ and azimuth ϕ. σ is 8° for the Gaussian lighting distribution.

We discretize the SH basis to the same size as our environment map. To get the SH coefficients, we project the environment map onto the basis function and integrate it over the surface of the unit

sphere. The formula of i-th coefficient is written as:

$$\alpha_i = \int_0^{2\pi} \int_0^{\pi} Y_i(\theta, \phi) I(\theta, \phi) sin\phi \, d\phi \, d\theta \tag{3.8}$$

where $Y_i(\theta, \phi)$ is the i-th SH basis function at the element of (θ, ϕ), $I(\theta, \phi)$ is the corresponding intensity value on environment map.

The discrete form is:

$$\alpha_i = \sum \sum Y_i(\theta, \phi) I(\theta, \phi) sin\phi \tag{3.9}$$

Ambient light: Since ambient light is inconsistent across images, especially between light stage and in-the-wild images, we introduce a method to estimate ambient light using the shadow mask. Since only ambient light contributes to shadowed regions, we use the average intensity of the shadow pixels as an estimate of the ambient light intensity in the image. To sum the contributions of directional and ambient light, we first model each image's directional light as a point light and estimate the corresponding 9 SH coefficients. We then add the estimated ambient light intensity to the 0th SH coefficient, which represents overall light intensity.

Since SH coefficients provide us with the overall global illumination without high-frequency details. Shadow mask is a necessary step to incorporate 3D geometric information and guide the neural network to learn how to remove and add shadows. We create shadow masks for all training images using the lighting direction and the 3D face mesh offered by 3DMM fitting. We estimate the 3D mesh, transformation matrix, and camera matrix for each face. To generate the shadow mask, we utilize ray tracing algorithm to cast parallel rays along the z-axis towards the mesh. If the ray hits the face, we check for two kinds of shadows at the intersection point: self shadows and cast shadows. Portions of the face between the light source and the intersection point will block the light and cast a shadow. To determine if the point lies in a cast shadow, we cast a ʻshadow feeler" ray from the intersection point to the light source [62]. If the ray hits the surface, the intersection point is in a cast shadow. To determine if the point lies in self shadow, we compute the angle between the light source's direction and the surface normal of the intersection point. If the angle is obtuse, the light hits the back of the surface and the kind of shadow is considered self shadow.

Method	Si-MSE	MSE	DSSIM
SfSNet [33]	0.0545	0.1330	0.3151
DPR [49]	0.0282	0.0702	0.1818
Our model	**0.0220**	**0.0292**	**0.1605**

Table 3.1 Relighting Performance Using Target Lighting on Multi-PIE. We compare our relighting performance against prior methods that take a source image and a target Spherical Harmonics lighting as input. Our method is able to outperform previous work across all three metrics.

Method	Si-MSE	MSE	DSSIM
Shih et al. [50]	0.0374	0.0455	0.2260
Shu et al. [53]	0.0162	0.0243	0.1383
Our model	**0.0148**	**0.0204**	**0.1150**

Table 3.2 Lighting Transfer Evaluation on Multi-PIE. We compare our lighting transfer performance against two existing lighting transfer algorithms. Each input image is assigned a random reference image. Our model outperforms both approaches across all three metrics.

If the intersection point is either in a cast shadow or self shadow, we assign 0 to the corresponding pixel. Otherwise, the pixel is illuminated and is 1 in the shadow mask.

Border weights: We reweight our ratio image estimation loss to assign larger weights to pixels near the shadow border. Higher-contrast hard shadows should have a higher weight than lower-contrast soft shadows, which pushes the neural network to put more emphasis on learning the shadow boundary accurately.

3.4 Result

3.4.1 Dataset

We train our model using images from two datasets, one with mostly diffuse lighting and one with strong directional lights. Our diffuse lighting dataset is the Deep Portrait Relighting (DPR) dataset, where we use the same training images as [49]. Our dataset with strong directional lighting is the Extended Yale Face Database B [63], which contains $16,380$ images of 28 subjects with 9 poses and 65 illumination conditions (64 distinct lighting directions and one ambient lighting).

(a) Source Image (b) Target Image (c) Our model (d) DPR [49] (e) SIPR [48] (f) SfSNet [33]

Figure 3.3 Qualitative Relighting Results on Multi-PIE Using Target Lighting. Here, we compare our relighting results on Multi-PIE with prior work that takes a source image and a target lighting as input. Images for SIPR [48] are provided by the authors. Notice that our model produces significantly better cast shadows, especially around the nose than previous methods.

(a) Source Image (b) Reference Image (c) Target Image (d) Our model (e) Shih et al. [50] (f) Shu et al.[53]

Figure 3.4 Qualitative Lighting Transfer Results on Multi-PIE. Here, we compare our lighting transfer results with two baselines by estimating the target lighting from the reference image. Images [53] are provided by the authors. Notice that our model is able to produce the correct lighting, whereas [50] produces the wrong lighting. Both baselines also strongly alter the skin tone of the subjects, whereas our method is able to maintain their skin tone reasonably well.

3.4.2 Quantitative Evaluation

We compare our model's relighting performance with prior face relighting work on the Multi-PIE dataset [64], where each subject is illuminated under 19 different lighting conditions (18 images with known directional lights, one image with no directional lighting).

Relighting Evaluation Using Target Lightings. We compare against prior relighting methods

(a) Source Image (b) Target Lighting (c) Our model (d) DPR [49] (e) SIPR [48] (f) SfSNet [33]

Figure 3.5 Qualitative Relighting Results on FFHQ Using Target Lighting. We compare our relighting results on FFHQ subjects with prior work. Images for SIPR [48] are provided by the authors. Across all lighting conditions, our model produces better cast shadows than prior work, especially around the nose and eyes.

[49, 33] that take a source image and a target lighting as input. For each subject and each session, we randomly select one of the 19 images as the source image and one image as the target image, which serves as the relighting groundtruth. The target image's lighting is then used to relight the source image. This leads to a total of 921 relit images. We evaluate the relighting performance using three error metrics: Si-MSE [49], MSE, and DSSIM. The results are shown in Table 3.1.

Relighting Evaluation Using Reference Images. We also compare our model's performance against relighting methods that require both a source and a reference image as input [50, 53]. These

methods relight the source image by transferring the reference image's lighting to the source. For each Multi-PIE image, we randomly select a reference image across all subjects from the dataset and estimate the target lighting from the reference image. We then relight the source image using the estimated target lighting. The results are shown in Table 3.2.

Facial Detail Preservation Evaluation. To compare our model's ability to preserve the subject's local and global facial details during relighting with prior work, we compute the average cosine similarity of the VGG-Face [65] features of the relit and groundtruth Multi-PIE images across different layers of VGG-Face. In particular, we use the layer before each max-pooling layer in VGG-Face. The results of the evaluation are shown in Fig. 3.6. Our model achieves noticeably higher cosine similarity than prior work in earlier layers and is comparable or slightly better than prior work in later layers, indicating that our model is better at preserving local facial features than previous methods.

3.4.3 Qualitative Evaluations

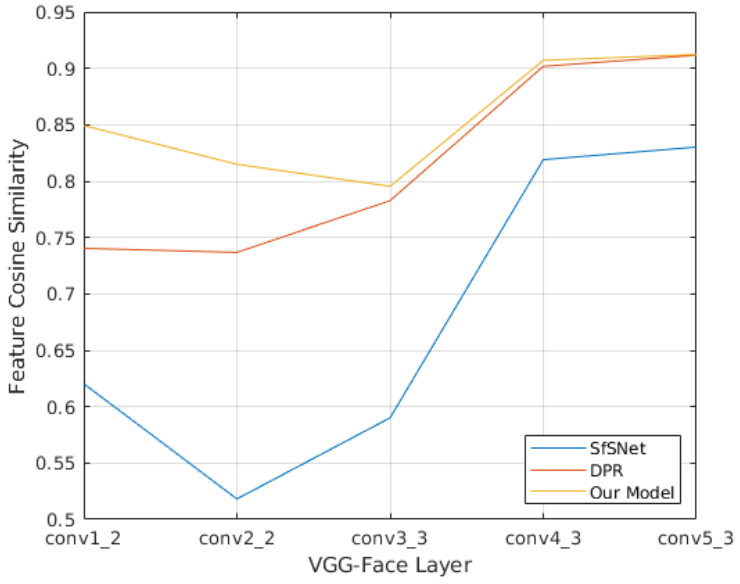


Figure 3.6 Facial Detail Preservation Evaluation on Multi-PIE. We compute the cosine similarity between the VGG-Face [65] features of each method's relit images and the ground truth target images as a measure of their ability to preserve the subject's local and global facial details during relighting. Notice that our method is consistently the best at preserving local features based on the cosine similarity in the earlier layers.

We demonstrate qualitative relighting results on the Multi-PIE [64] dataset and the FFHQ dataset [66] and compare them with prior work.

On Multi-PIE, we include relit images using target lighting (See Fig. 3.3) as well as relit images produced from lighting transfer (See Fig. 3.4). When applying target lighting, our model is able to produce noticeably better cast shadows than DPR [49], SIPR [48], and SfSNet [33]. Our model also avoids overlighting the image, whereas [49] often produces images that appear overexposed. When performing lighting transfer, Shih et al. [50] is unable to produce the correct lighting whereas our model is able to estimate and transfer the correct target lighting.

On FFHQ, we perform relighting using target lighting and compare it with previous approaches (See Fig. 3.5). Our approach handles cast shadows better than prior work, as seen by the shadows around the nose, eyes, and other regions, while also maintaining similar or better visual quality in the relit images.

CHAPTER 4

NERF HEAD

4.1 Introduction

Building free-view human head avatars is a long-standing research topic in the computer vision, computer graphics, and machine learning communities. Such avatars are crucial in various VR/AR applications, in particular, in facial animation for teleconferencing, social media apps, movie/animation production, and game industry. One common approach in previous methods is to use 3D morphable models (3DMM) to approximate the human head geometry and thus provide access to head poses and facial expressions. The deformation of the 3DMM is often driven by a low dimensional parameter space such as PCA-based features. By changing the parameters of the expression/pose/identity in this space, we could articulate the 3D template shape with characteristics of specific individuals.

However, the 3D reconstruction based on 3DMM usually lacks details even with the appearance modeled through texture mapping, due to the limitation in resolution and the restriction in feature space dimension for template-based 3D explicit representations that are efficient enough for real-time rendering. The model is typically merely capable of building a head mesh without hair or more often just a face mesh. It could not model hair and head accessories with complex geometric and topological features, like glasses and earrings, which are typically present in real life. Moreover, it remains a challenging problem to separately generate photorealistic view-dependent textures for 3DMM shapes, leading to difficulties in realistic rendering of human head images using simple illumination models. A concurrent work, Neural Head[67], creates a relatively accurate dynamic mesh including the explicit geometric modeling of hair and accessories along with the corresponding textures. However, rendering quality of the resulting model-based images seems still lower than that of implicit models with image-based rendering when sufficient input images are provided for training.

Image-based rendering is another major approach for rendering generic digital scenes, including human heads, which can be carried out without any explicit mesh-based geometric models. For

instance, implicit volumetric representations such as Neural Radiance Fields (NeRF)[68], have been successfully employed for rendering various scenes with increasing popularity. Radiance fields can sample the radiance value at an arbitrary spatial point along an arbitrary direction within a certain range, thus allowing assembly output images from any camera location with any view direction within the range. In particular, NeRF learns an implicit volumetric density and RGB color field by fitting the rendered output to given input images from different views. With the implicit neural radiance field, one may directly generate photo-realistic renderings by assembling rays from the projection center to the image plane of any virtual camera, without any geometric modeling of object surface or a texture mapping. The perceived geometric and appearance details of NeRF results are hard to match for explicit methods like 3DMM. Moreover, some NeRF models also offers realistic relighting.

There are a number of existing works that investigated the possibility of applying NeRF to render humans. For instance, EG3D[69] used three orthogonal planes to create an efficient NeRF representation, but it is not straightforward to generalize the model to handle the dynamic case when the face is deforming. Some approaches [70, 71] tried to combine NeRF with 3DMM parameters, but without the 3D shape guidance, they require a relatively large number of input images to learn the 3D neural field.

Inspired by Neural Body[72] (which use NeRF to model full-body animation under the guidance of a deforming mesh) we propose a hybrid approach that combines the mesh representation with an implicit volumetric rendering to create photo-realistic results. This is done through a latent code stored on a dynamic mesh driven by expression and pose parameters, with the deformable shape embedded in an ambient volume. In contrast to Neural Body input, which has multiple views (although sparse), we present a method for face avatar that may take single view input and is able to reduce the input frames required for training significantly.

Our main contributions include:

- a 3DMM mesh-based NeRF pipeline for faces;

- a NeRF-based 3DMM mesh perturbation for refined NeRF; and
- a double-layer mesh method for better code diffusion for model-guided NeRF.

4.2 Related Work

4.2.1 View synthesis and neural scene representation

Traditional image-based rendering methods [73, 74] often use radiance/light field interpolation to generate novel views. Such methods often offer a limited range of viewpoints and require extremely dense input views for high quality output. As deep learning technology advances, learnable image-based methods quickly gained popularity. Such methods warp and synthesize input images into novel views[75] through sparser input data. Nevertheless, learning-based view synthesis still requires a large amount of image data and lacks the understanding of the underlying 3D geometry. To make full use of information from 3D scenes, some methods [76] utilize depth information to guide the synthesis of novel views. They, instead, rely heavily on the geometry proxy estimated from the input views. There are also methods[77] that aim to build 3D mesh representations explicitly, but accurate generic 3D reconstruction itself remains an open problem so far. A template mesh with fixed topology is typically unavailable for generic 3D scenes in the real world. Even generating photo-realistic textures for explicit meshes can be challenging, and the rendering with textured meshes still deviates further from the real-world images compared to purely image-based rendering. Recent methods [78, 79, 80, 81] that use implicit representations combined with neural network seem to finally meet the requirements of high-quality view synthesis. They model the radiance fields through volumetric fields of color and density (i.e., one set of RGB values and one density value at each point or voxel), given only a set of input RGB images. NeRF[68] attained great success in embedding feature volumes with neural networks and generate impressive results by outputting radiance given specific locations and directions through volumetric rendering. Our method follows the geometry-guided NeRF framework but incorporates a component that uses NeRF to reversely guide the geometry, and reiterate the NeRF generation to refine the results.

4.2.2 NeRF-based dynamic scene representation

The original NeRF is designed for static scenes. However, follow-up works like [82, 83, 71, 84, 72] extend NeRF to dynamic scenes by taking time component as part of the input. [82, 83] uses scene flow to linearly interpolate between different frames. [84, 71, 85] convert different temporal frames to a canonical frame. Specifically, [71, 85] model human head with the assumption that head motion is small in typical input and output deformation/animation. [70, 72] utilizes a global transformation matrix to embed dynamic NeRF in a canonical space to address large movements in the video. [70] conditions NeRF on 3DMM expression parameters for different frames, while [72] uses SMPL parameters based on a template human body mesh to integrate different frames. We follow the embedding-based method for NeRF generation, but added a double-layer embedding to speed up the diffusion of the implicit model.

4.2.3 Human face reconstruction

3DMM is typically driven by a low-dimensional parameter space. Most face reconstruction methods [86] reply on 3DMM to provide a coarse template mesh as initialization. They either build the blendshape basis for high-frequency details like fine wrinkles [87] or learn the non-rigid deformation on top of the coarse surface[67]. NeRF-based methods[70, 69] learn an implicit representation for the human head. To build the explicit mesh, EG3D[69] extracts the isosurface from the depth field, which is technically only a 2.5D mesh. Our NeRF output can also be used to generate a 2.5D mesh to provide an approximate geometry if needed. Moreover, we allow a more flexible mesh deformation directly guided by the NeRF density field without resorting to a blendshape basis.

4.2.4 Human avatar reconstruction

Traditional methods are usually based on 3DMM parameters to reanimate human subjects. 3DMM models built on a large of blendshapes provide us with the shape variation on a learned shape space. Therefore, the deformation could be limited and the control over the template mesh is coarse-grained. Recent methods start to combine 3DMM with neural network. [88, 86] disentangle appearance and shape. [88] use shape parameter to model deformation, while color

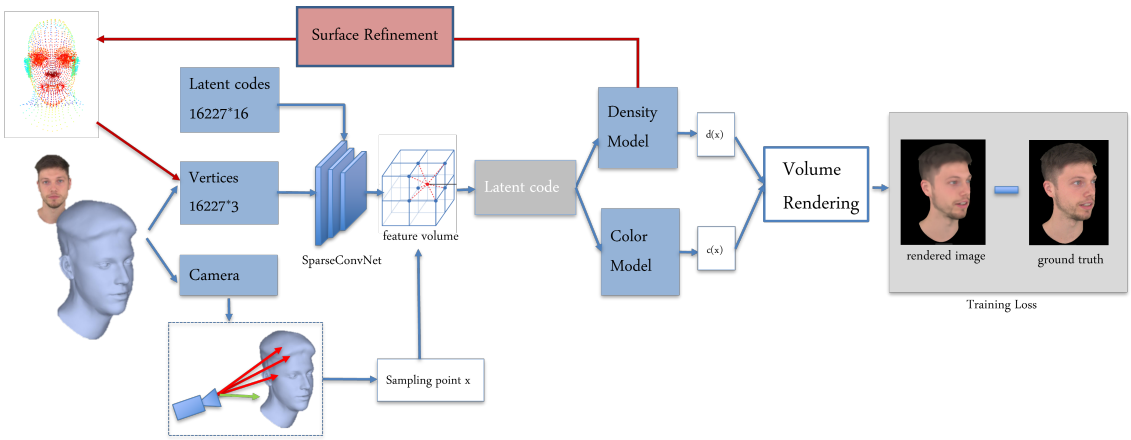


Figure 4.1 Overview of our proposed method.

map in canonical space stays unchanged. [86] assign feature descriptors to each surface vertex to learn texture synthesis for the deformed mesh. In addition, the use of GAN enables the generation of more photo-realistic rendering for human avatars, like in [89]. [72, 90] use one network to learn the detailed shape deformation based on 3DMM face reconstruction while utilizing neural rendering to recover the texture information. 3DMM parameters are utilized to generate novel poses and expressions. Usually this kind of methods focus on mesh reconstruction quality, and the rendered image quality is not comparable to NeRf-based methods. To gain fine-grained control over avatar attributes, [91] uses local face models instead of 3DMM to offer more flexibility and expressiveness, and [92] represents attributes as localized masks and regresses each attribute value and its corresponding mask using neural network. It requires 2D mask annotations for a few shots that specify which region of the image an attribute controls.

We design our avatar algorithm based on NeRF, the neural radiance field enables us to render high-fidelity face images. Inserting the human head mesh in the 3D neural field enables our network to reconstruct a dynamic human head. Additionally, the 3DMM parameters provide us with control over pose and expression. Finally, our NeRF-based rendering pipeline could learn the high-frequency details, such as hair and glasses, on top of the rough shape deformation without annotations.

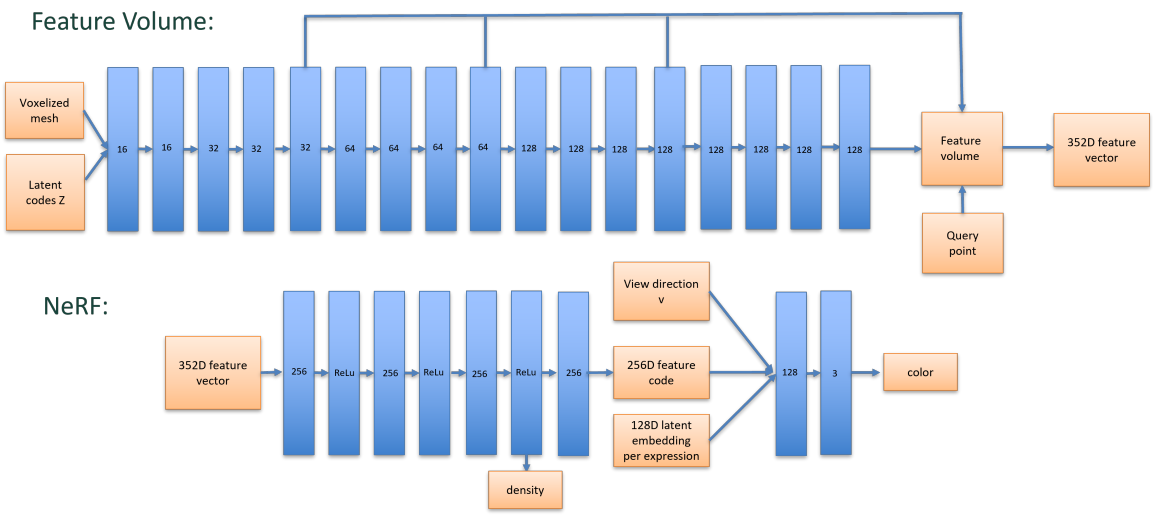


Figure 4.2 Network structure.

4.3 Algorithm

Given a single-view portrait video, our method reconstructs a dynamic human head avatar which enables the novel view synthesis of each input scene and arbitrary control of facial expressions and head poses. Our model learns an implicit neural representation based on NeRF[68] for human head. The overall pipeline is illustrated in Fig. 4.1. For each frame of a portrait video, we first reconstruct the coarse 3d morphable mesh for each frame. We use DECA[87] to also extract the corresponding pose, expression, and camera parameters for each frame. The pose parameters allow us to apply rigid global transformation to embed the head shape in a canonical bounding box which is shared by all frames. The expression parameter could be used for future facial expression manipulation. Inspired by [72], we attach the latent code for NeRF generation on the coarse 3D shape, whose embedding allows the generation of dynamic density and RGB fields. The latent codes attached to each vertex on the head surface typically denote the local geometric deformation and texture information during the training of the neural network. To generate the continuous radiance field, the latent codes are diffused to nearby space through a code diffusion process. The neural network could decode the sparse volumetric latent code field into color and density fields. The output image is then generated from different viewpoints via volume rendering.

4.3.1 Coarse facial geometry

For this part of our pipeline, we simply employ DECA [87]. We provide the details on input preparation and our parameter choices below. Given single-view or multi-view portrait images/videos, we first randomly choose input frames that may cover a wide enough range of view angles if possible, since we aim to generate novel views of portrait animation. Then, for each input frame, which may have different expressions and poses, we use DECA [87] to reconstruct the shape S_t, a parameterized mesh of 5,023 vertices. The type of coarse output mesh in DECA is FLAME[93], a statistical 3D head mesh model that combines separate linear identity shape and expression spaces with linear blend skinning (LBS) to articulate the neck, jaw, and eyeballs. The advantage of the consistent connectivity across different frames is the straightforward deformation of the NeRF volume fields as we detail next.

We optionally allow enhancement for different regions of the face, for instance allowing the forehead area or top of the head to have two layers of vertices by offsetting the original mesh vertex along the normal direction by a preset distance. Similar treatment can be performed near the eyes or mouth for enhanced details to accommodate accessories or expressions. While our ablation study showed no improvements in the statistics that we employ for accuracy, the improvements to the targeted regions are visually noticeable.

4.3.2 Dynamic feature volume

A set of latent codes $Z = \{z_1, z_2, ..., z_{5023}\}$ are anchored to the surface of shape S_t, each latent code is a 16 dimensional vector to encode local appearance information and geometric details on top of the underlying 3D shape. The same vertex across all the input frames share the same set of latent codes, which integrate all the frames together and take advantage of the consistency across different frames in terms of the underlying 3D shape and the facial texture. The dynamics brought by expression parameters affect the 3D shape S_t directly, therefore the embedding locations of the latent code in the feature volume will change accordingly, which leads to a dynamic feature volume to allow expression-dependent output. While our model has been trained to be individual-specific due to data availability, we believe it is possible to build feature volumes that also takes identity

parameters for generic human avatar, since both the coarse geometry and appearance model can be used to train the latent code representation across different identities.

Following[72, 94], we use SparseCovNet [95] to take S_t and Z as input and generate a multi-scale feature volume. As a typical NeRF field is represented by continuous volumetric fields instead of samples on a dynamic surface, SparseCovNet provides a reasonable bridge to link the sparse surface points to the ambient space with an implicit learned diffusion-like process. With this conversion, given a query point, we may retrieve its latent code in the multi-scale feature volume as shown in Fig. 4.2 and concatenate all the latent code at different scales as the final 352D feature vector. The latent code is decoded in the next stage of our pipeline into density and color functions.

More specifically, we first compute the bounding box of the 3D head shape S_t, and then divide the bounding box to a cartesian grid. Each grid cell (voxel) has a size of $5mm \times 5mm \times 5mm$. The latent code assigned to each voxel is calculated as the mean of the latent codes on all the surface points situated inside the voxel. Similar to [72, 94], we also downsample the feature volume by the factors of 2, 4, 8 and 16 to build multi-scale feature volumes. From the feature volume at each scale, we obtain a feature vector at each query point through trilinear interpolation, and then concatenate all four feature vectors into a 352-dimensional vector as final output. For frame t, We denote the feature vector at a given sample point x. The output at this stage is thus a function of the sample point location x, parameterized by the latent code L attached to the mesh vertices, and the given head shape S_t, written as $f(x; S_t, Z)$.

While this embedding process is fairly standard, we optionally provide a NeRF-guided deformation to the surface point locations. Thus, after the first iteration of training, we perturb the position of each vertex to get a more accurate mesh surface, then insert the new surface points back to the 3D feature volume, and train the network again to generate new rendering images. In our ablation study in the next section, we found mixed results for this refinement. We speculate that we may need larger training sets or more accurate initial geometry to consistently outperform our current pipeline.

4.3.3 Color/density estimation

At this stage of our pipeline, the feature vector associated with a spatial location is passed into the MLP network to regress density and color at this point. For color regression, we add viewing direction as input, since without viewing direction, the model will have difficulty representing specularity. We also add another per-frame 128D latent code to encode the temporally-varying factors across frames, such as illumination conditions. Given this treatment, the radiance field is indeed a 5D function that depends both on location and direction.

Specifically, given a feature volume and a viewing direction, the radiance value at a given viewpoint is determined by casting a ray from the viewpoint along the viewing direction towards the feature volume. If the ray intersects with the bounding volume, sample points are distributed along the portion of the ray that is inside the bounding volume. For each sample point, we query the multi-scale feature volume to assemble the 352D latent code, which is then fed to a multi-layer perceptron (MLP) network along with the viewing direction to estimate the output color and density value, as done in typical practice.

As the final representation of our dynamic radiance field, our MLP network takes the feature vector as as input, and outputs color and density predictions for each sample point inside the bounding volume. For a given frame t, the density is defined as a function of latent code at sample point, parameterized by a per-frame latent code that takes into account additional changes in the environment, such as illumination.

Density The density model is trained to be directly a function of only the latent code $L(x, Z, S_t)$ at a given location x, the given latent code Z for the entire mesh, and the current mesh shape S_t:

$$d(x) = M_d(L(x, Z, S_t)), \tag{4.1}$$

where M_d is the MLP network for density as shown in Fig. 4.2.

Color The color model additionally takes the given viewing direction v and the 128D latent code l_t associated with frame t along with latent code $L(x, Z, S_t)$ as its three input:

$$c(x) = M_c(L(x, Z, S_t), v, l_t), \tag{4.2}$$

where M_c is the MLP network for color as shown in Fig. 4.2.

4.3.4 Volumetric rendering with radiance fields

Accumulating the data along each ray and assembling rays into images is a standard procedure in volume rendering[96]. Once we obtain the procedure to produce radiance associated with any ray, the image is constructed by a simple loop that iterates over all the pixels. Thus, we provide the details for radiance estimate based on the sample points in the previous stage. We may see it as the evaluation of the radiance field based on the internal per-point density and directional emitted radiance (in RGB color) representation.

For the output color (which corresponds to the direction-dependent physical quantity radiance), we integrate the color and density of the sample points along the ray. Specifically, the color obtained in the previous stage encodes the radiance emitted from one location towards the given direction, and the density indicates how the light interacts with the matter in the small volume at the point, filtering out a fraction of the light passing through the volume. A practical way to perform the integral numerically is the approximation through quadrature. Thus, based the previous sampled point data, the final color of the corresponding ray r that reaches the projection center (origin of r) of the camera is given by:

$$C(r) = \sum_{k=1}^{N} T_k(1 - \exp(-d(\mathrm{x_k})\delta_k))c(\mathrm{x_k}), \tag{4.3}$$

where

$$T_k = \exp(-\sum_{j=1}^{k-1} d(\mathrm{x_j})\delta_j), \tag{4.4}$$

and $\delta_k = ||\mathrm{x_{k+1}} - \mathrm{x_k}||_2$ is the distance between adjacent sampled points, $d(x_k)$ is the density at sample point x_k, $c(x_k)$ is the color at the sample point x_k along the ray direction. We set the total sample point number N to 64 in our experiment. Note that the sample points are sorted from near to far.

4.3.5 3D mesh extraction

Some 3D surface reconstruction methods exist for extracting geometric information from neural radiance fields generated by MLP. However, methods like EG3D [69] only generate a 2.5D mesh, i.e., a depth map from a given camera direction. Since the internal density model of NeRF learns for every point a probability of whether it is inside, it is only accurate in the sense of reproducing the image rendering. Thus, it is not necessarily reliable for the extraction of an accurate surface. On the other hand, the integral data along rays, as done in volumetric rendering, is more accurate than the point samples. That is why the 2.5D meshes generated by integrating a chosen direction are often more stable since they are based on the reasonable estimate of the closest point of the scene to the camera along each projection ray.

However, to refine the mesh used in the NeRF construction, we need an honest 3D mesh. Thus, we propose to keep the mesh connectivity intact, and only perturb the locations of existing vertices. To have a reliable estimate as in the 2.5D reconstruction, we mimic the procedure of volumetric rendering by tracing along rays. The difference is that now we construct a ray that passes through the initial surface vertex locations, with the ray direction following the inward-pointing normal direction at that point of the original surface.

To ensure that we start from an outside point, we first move surface point V to the outside space of the head through offsetting the point by a safe distance a. Similar to 2.5D depth image generation, we calculate the depth for the point along the negative normal direction, and then move the surface point back along the negative normal direction by distance d to reach the surface point. The new location V' of the surface point is calculated as follows:

$$V' = V + (a - d)\hat{n}, \tag{4.5}$$

where $\hat{n}$ is the unit outward normal of the surface mesh. To make sure that the mesh is only perturbed, we clamp the $a - d$ to the range $[-\epsilon, \epsilon]$, where $\epsilon = 5mm$ is the offset distance upper bound.

The following is the details of our procedure to generate depth for each sample point by integrating density along the normal direction to avoid the drawback of a 2.5d depth map: First, we

select a large a (0.03 in our experiment) to make sure we reach the exterior of the human head, the density will be effectively zero in the outside space. Newt, we select a negative distance b(-0.01 in our experiment), to make sure we reach the interior of the head, where the density will be nearly 1. Then, we query the density function for 64 sample points between $V + a\hat{n}$ and $V + b\hat{n}$. For each sample point, we integrate the density between this point and $V + a\hat{n}$:

$$depth = \sum_{i=1}^{N} T_i(1 - e^{-\delta_i \sigma_i})d_i, \tag{4.6}$$

where T_i is evaluated the same way as in the volumetric rendering.

The overall algorithm to extract a perturbed mesh from the neural radiance field is outlined as:

Algorithm 4.1 Surface Update

```
function SurfaceUpdate(V, n̂, a, b)
    for i = 1 to n do                                   ▷ n = 5,023: vertex count
        sample 64 points P along n̂_i with an offset between a and b
        for k = 1 to 64 do
            depth(P_k) = Σ_k T_k(1 − e^{−δ_k σ_k}) d_k
            if depth(P_k) == 0 then
                d = location(P_k) break
            end if
        end for
        clamp a − d to [−ε, ε]
        V'_i = V_i + (a − d) n̂_i
    end for
end function
```

4.3.6 Loss function

There are several choices in loss functions that we have tested. The differences are the treatments of background pixels. The simplest choice is to model the background within the bounding volume, it creates the best transition but typically distorts the background in novel views. Another choice is through using automated background masks, we ignore the pixels in the background region in the loss function. This allows the network to focus on the foreground, but would require the use of the roughly estimated foreground mask based on the DECA mesh. The choice that creates better geometric construction is to force background pixels to match an accumulated alpha value of 0, which forces the density to appear where the head intersects with the camera rays. Regardless of

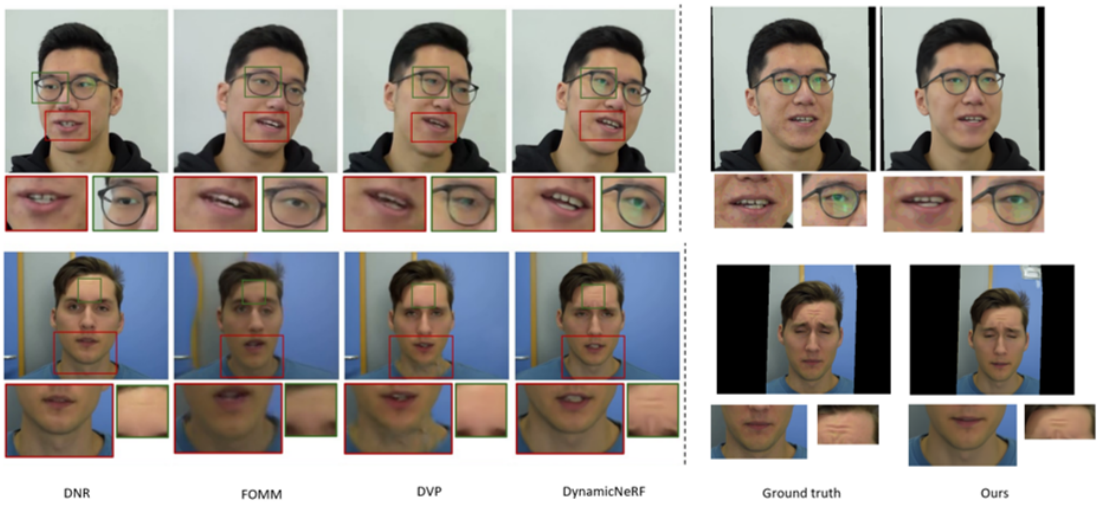


Figure 4.3 Comparison to state-of-the-art algorithms on self-reenactment.

the choice, we optimize the network to reduce the difference between the rendered image and the ground truth images, with the loss function expressed as:

$$L = \frac{1}{N} \sum_{r \in \mathbf{R}} ||C(r) - C_t(r)||_2^2 \tag{4.7}$$

where $\mathbf{R}$ is the set of camera rays that pass through the image plane, N is the number of sampled rays, and C_t is the ground truth pixel value. The summation range is according to the background treatment choice.

4.4 Experiments

We tested our pipeline on multiple portrait videos from a diverse set of sources. We get three human videos by the courtesy of NerFace, two by the courtesy of authors of Neural Head, and two by the courtesy of authors of CoNeRF. In additional we downloaded public-domain videos of Barack Obama and Michelle Obama from Youtube to demonstrate the generality and robustness of our model. Since NerFace training dataset is already shuffled, we randomly select 300 frames from the test dataset for testing. For CoNeRF, we use 300 frames for training and take every other frame for testing.

We measure the typically used metrics: L_1 distance, Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS), and report

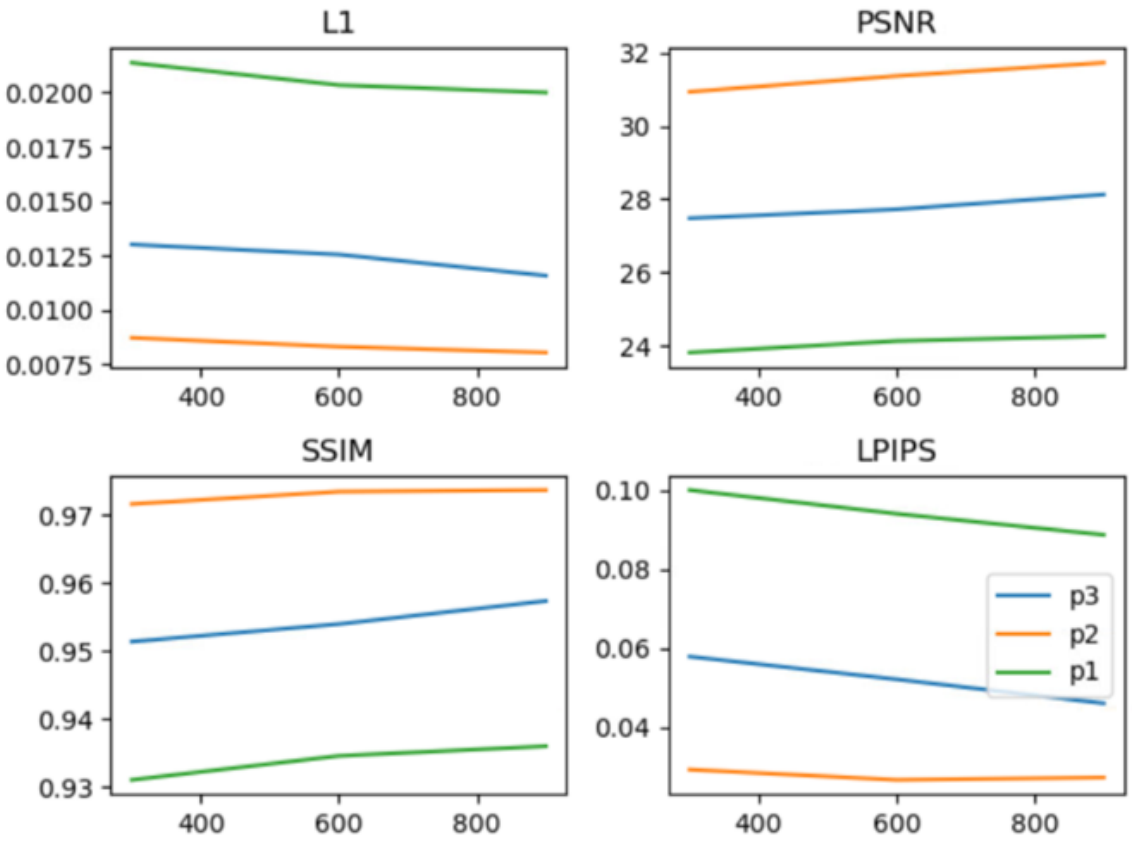


Figure 4.4 Increasing training frames will increase the performance on all four metrics.

Video	$L_1 \downarrow$	PSNR↑	SSIM↑	LPIPS↓
Head0	0.012	26.49	0.94	0.11
Head4	0.011	27.12	0.95	0.08
Person1	0.021	23.81	0.93	0.10
Person2	0.010	30.23	0.97	0.03
Person3	0.013	27.48	0.95	0.06
Video1	0.026	22.56	0.91	0.09
Average	0.015	26.28	0.94	0.08

Table 4.1 Quantitative result on different videos.

them in Table 4.4. for different videos, the resolution, frame rate, and human head size in the video differ significantly, which may influence the performance of our algorithm. We also compare our quantitative results with baselines, like NerFace, FOMM, and DVP. Our algorithm outperforms these baselines. As for conerf, it requires users to manually annotate facial attributes. Our method and conerf don't fall into the same category, there is no need for direct comparison with conerf.

Figure 4.5 The second iteration could capture more high-frequency details in mouth and glass region.

Method	$L_1 \downarrow$	PSNR↑	SSIM↑	LPIPS↓
FOMM	0.036	23.77	0.91	0.16
DVP	0.021	25.67	0.93	0.10
NerFace	0.029	24.22	0.93	0.09
CoNeRF	0.015	32.24	0.98	0.17
Average	0.015	26.28	0.94	0.08

Table 4.2 Comparison with baselines.

Method	$L_1 \downarrow$	PSNR↑	SSIM↑	LPIPS↓
1st iteration	0.014	27.26	0.95	0.06
2nd iteration	0.014	27.45	0.95	0.06

Table 4.3 Ablation study.

The quantitative comparison is shown in table 4.4. The qualitative comparison is shown in Fig. 4.3. As can be seen from the qualitative result, our algorithm could reconstruct the appearance and facial expressions and preserve facial details including wrinkles, while other algorithms either did not reconstruct correct facial expressions or missed high-frequency details, in these test cases.

We ran an ablation study on the three videos from Neural Head to investigate what could be

the key factor to influence the final performance. Increasing the training frames will significantly increase the accuracy until a certain minimum frame number is reached, as shown in Fig 4.4. Our ablation study in table 4.4 also shows the refined mesh improves the overall performance slightly. However, from the qualitative result in Fig. 4.5, we could see in some specific circumstances, the refined mesh could improve the reconstruction of local regions, like the more deformable mouth region and the glasses that are not in the original DECA mesh. Another reason may have been that since DECA does not reconstruct the mouth cavity when the mouth is open, our refined mesh would assign vertices in this region, and the feature vectors attached to the vertices will encode appearance information and increase the expressiveness of our algorithm.

4.5 Conclusion and Future Work

We presented a 3DMM-based NeRF for dynamic human head rendering. With the guidance of geometric shapes, we could significantly reduce the training frames. We could generate high-fidelity novel-view face images and also reanimate the human head by changing 3DMM parameters. While the refined mesh does not improve the result further, it shows promise in its handling of local regions. We will explore the use of shape regularizers and region-specific local distance thresholding, as well as seek a more accurate initial 3D mesh, to improve our neural radiance field-guided mesh perturbation. There are other aspects of our pipeline that could have been enhanced to improve the final rendering performance, such as the use of better foreground mask generation.

BIBLIOGRAPHY

[1] N. Lian, X. Wang, Y. Jing, and J. Lin, "Regulation of cytoskeleton-associated protein activities: Linking cellular signals to plant cytoskeletal function," *Journal of Integrative Plant Biology*, vol. 63, no. 1, pp. 241–250, 2021.

[2] L. Blanchoin, R. Boujemaa-Paterski, C. Sykes, and J. Plastino, "Actin dynamics, architecture, and mechanics in cell motility," *Physiological reviews*, vol. 94, no. 1, pp. 235–263, 2014.

[3] G. J. Brouhard and L. M. Rice, "Microtubule dynamics: an interplay of biochemistry and mechanics," *Nature reviews Molecular cell biology*, vol. 19, no. 7, pp. 451–463, 2018.

[4] P. Li and B. Day, "Battlefield cytoskeleton: turning the tide on plant immunity," *Molecular Plant-Microbe Interactions*, vol. 32, no. 1, pp. 25–34, 2019.

[5] P. Nick, "Mechanics of the cytoskeleton," in *Mechanical integration of plant cells and plants*, pp. 53–90, Springer, 2011.

[6] J. G. Carlton, H. Jones, and U. S. Eggert, "Membrane and organelle dynamics during cell division," *Nature Reviews Molecular Cell Biology*, vol. 21, no. 3, pp. 151–166, 2020.

[7] I. Kristó, I. Bajusz, C. Bajusz, P. Borkúti, and P. Vilmos, "Actin, actin-binding proteins, and actin-related proteins in the nucleus," *Histochemistry and cell biology*, vol. 145, no. 4, pp. 373–388, 2016.

[8] M. Melak, M. Plessner, and R. Grosse, "Actin visualization at a glance," *Journal of cell science*, vol. 130, no. 3, pp. 525–530, 2017.

[9] N. Lichtenstein, B. Geiger, and Z. Kam, "Quantitative analysis of cytoskeletal organization by digital fluorescent microscopy," *Cytometry Part A: The Journal of the International Society for Analytical Cytology*, vol. 54, no. 1, pp. 8–18, 2003.

[10] S. A. Shah, P. Santago, and B. K. Rubin, "Quantification of biopolymer filament structure," *Ultramicroscopy*, vol. 104, no. 3-4, pp. 244–254, 2005.

[11] Y. Liu, A. Nedo, K. Seward, J. Caplan, and C. Kambhamettu, "Quantifying actin filaments in microscopic images using keypoint detection techniques and a fast marching algorithm," in *2020 IEEE International Conference on Image Processing (ICIP)*, pp. 2506–2510, IEEE, 2020.

[12] Y. Liu, K. Mollaeian, and J. Ren, "An image recognition-based approach to actin cytoskeleton quantification," *Electronics*, vol. 7, no. 12, p. 443, 2018.

[13] M. Alioscha-Perez, C. Benadiba, K. Goossens, S. Kasas, G. Dietler, R. Willaert, and H. Sahli, "A robust actin filaments image analysis framework," *PLoS computational biology*, vol. 12,

no. 8, p. e1005063, 2016.

[14] T. Higaki, N. Kutsuna, T. Sano, N. Kondo, and S. Hasezawa, "Quantification and cluster analysis of actin cytoskeletal structures in plant cells: role of actin bundling in stomatal movement during diurnal cycles in arabidopsis guard cells," *The Plant Journal*, vol. 61, no. 1, pp. 156–165, 2010.

[15] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[16] W. Niblack, *An introduction to digital image processing*. Strandberg Publishing Company, 1985.

[17] T. Higaki, K. Akita, and K. Katoh, "Coefficient of variation as an image-intensity metric for cytoskeleton bundling," *Scientific reports*, vol. 10, no. 1, pp. 1–13, 2020.

[18] T.-C. Lee, R. L. Kashyap, and C.-N. Chu, "Building skeleton models via 3-d medial surface axis thinning algorithms," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 6, pp. 462–478, 1994.

[19] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 317–324, 1999.

[20] J. Nunez-Iglesias, A. J. Blanch, O. Looker, M. W. Dixon, and L. Tilley, "A new python library to analyse skeleton images confirms malaria parasite remodelling of the red blood cell membrane skeleton," *PeerJ*, vol. 6, p. e4312, 2018.

[21] K. Tanaka, S. Takeda, K. Mitsuoka, T. Oda, C. Kimura-Sakiyama, Y. Maéda, and A. Narita, "Structural basis for cofilin binding and actin filament disassembly," *Nature communications*, vol. 9, no. 1, pp. 1–12, 2018.

[22] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[23] G. W. Zack, W. E. Rogers, and S. A. Latt, "Automatic measurement of sister chromatid exchange frequency.," *Journal of Histochemistry & Cytochemistry*, vol. 25, no. 7, pp. 741–753, 1977.

[24] C. Li and P. K.-S. Tam, "An iterative algorithm for minimum cross entropy thresholding," *Pattern recognition letters*, vol. 19, no. 8, pp. 771–776, 1998.

[25] J.-C. Yen, F.-J. Chang, and S. Chang, "A new criterion for automatic multilevel thresholding," *IEEE Transactions on Image Processing*, vol. 4, no. 3, pp. 370–378, 1995.

[26] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern recognition*, vol. 33, no. 2, pp. 225–236, 2000.

[27] J. L. Henty-Ridilla, J. Li, B. Day, and C. J. Staiger, "Actin depolymerizing factor4 regulates actin dynamics during innate immune signaling in arabidopsis," *The Plant Cell*, vol. 26, no. 1, pp. 340–352, 2014.

[28] J. L. Henty-Ridilla, M. Shimono, J. Li, J. H. Chang, B. Day, and C. J. Staiger, "The plant actin cytoskeleton responds to signals from microbe-associated molecular patterns," *PLoS pathogens*, vol. 9, no. 4, p. e1003290, 2013.

[29] M. Guo, P. Kim, G. Li, C. G. Elowsky, and J. R. Alfano, "A bacterial effector co-opts calmodulin to target the plant microtubule network," *Cell host & microbe*, vol. 19, no. 1, pp. 67–78, 2016.

[30] M. Shimono, Y.-J. Lu, K. Porter, B. H. Kvitko, J. Henty-Ridilla, A. Creason, S. Y. He, J. H. Chang, C. J. Staiger, and B. Day, "The pseudomonas syringae type iii effector hopg1 induces actin remodeling to promote symptom development and susceptibility during infection," *Plant physiology*, vol. 171, no. 3, pp. 2239–2255, 2016.

[31] Y.-J. Lu, P. Li, M. Shimono, A. Corrion, T. Higaki, S. Y. He, and B. Day, "Arabidopsis calcium-dependent protein kinase 3 regulates actin cytoskeleton organization and immunity," *Nature communications*, vol. 11, no. 1, pp. 1–12, 2020.

[32] H. Le and I. Kakadiaris, "Illumination-invariant face recognition with deep relit face images.," in *WACV*, IEEE, 2019.

[33] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs, "SfSnet: Learning shape, refectance and illuminance of faces in the wild.," in *CVPR*, IEEE, 2018.

[34] T. Nestmeyer, J.-F. Lalonde, I. Matthews, and A. Lehrmann, "Learning Physics-guided Face Relighting under Directional Light.," in *CVPR*, IEEE, 2020.

[35] Y. Wang, L. Zhang, Z. Liu, G. Hua, Z. Wen, Z. Zhang, and D. Samaras, "Face relighting from a single image under arbitrary unknown lighting conditions.," *PAMI*, 2009.

[36] B. Egger, S. Schonborn, A. Schneider, A. Kortylewski, A. Morel-Forseter, C. Blumer, and T. Vetter, "Occlusion-aware 3d morphable models and an illumination prior for face image analysis.," *IJCV*, 2018.

[37] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras, "Neural face editing with intrinsic image disentangling.," in *CVPR*, IEEE, 2017.

[38] L. Tran and X. Liu, "Nonlinear 3d face morphable model.," in *CVPR*, IEEE, 2018.

[39] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. T. Freeman, "Unsupervised training for 3d morphable model regression.," in *CVPR*, IEEE, 2018.

[40] A. Tewari, M. Zollofer, H. Kim, P. ¨ Garrido, F. Bernard, P. Perez, and T. Christian, "Mofa: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction.," in *ICCV*, IEEE, 2017.

[41] S. Yamaguchi, S. Saito, K. Nagano, Y. Zhao, W. Chen, K. Olszewski, S. Morishima, and H. Li, "High-fidelity facial reflectance and geometry inference from an unconstrained image.," *ACM Transactions on Graphics*, 2018.

[42] J. T. Barron and J. Malik, "Shape, illumination, and reflectance from shading.," *TPAMI*, 2015.

[43] D. Shahlaei and V. Blanz, "Realistic inverse lighting from a single 2d image of a face, taken under unknown and complex lighting.," in *International Conference on Automatic Face and Gesture Recognition*, IEEE, 2015.

[44] J. Lee, R. Machiraju, B. Moghaddam, and H. Pfister, "Estimation of 3D faces and illumination from single photographs using a bilinear illumination model.," in *Eurographics Conference on Rendering Techniques*, 2005.

[45] J. Lin, Y. Yuan, T. Shao, and K. Zhou, "Towards High-Fidelity 3D Face Reconstruction from In-the-Wild Images Using Graph Convolutional Networks," in *CVPR*, IEEE, 2020.

[46] G.-H. Lee and S.-W. Lee, "Uncertainty-Aware Mesh Decoder for High Fidelity 3D Face Reconstruction," in *CVPR*, IEEE, 2020.

[47] C. Li, K. Zhou, and S. Lin, "Intrinsic face image decomposition with human face priors.," in *ECCV*, IEEE, 2014.

[48] T. Sun, J. T. Barron, Y.-T. Tsai, Z. Xu, X. Yu, G. Fyffe, C. Rhemann, J. Busch, P. Debevec, and R. Ramamoorthi, "Single image portrait relighting.," *ACM Transactions on Graphics (SIGGRAPH)*, 2019.

[49] H. Zhou, S. Hadap, K. Sunkavalli, and D. W. Jacobs, "Deep single-image portrait relighting," in *ICCV*, IEEE, 2019.

[50] Y. Shih, S. Paris, C. Barnes, W. T. Freeman, and F. Durand, "Style transfer for headshot portraits.," *ACM Transactions on Graphics*, 2014.

[51] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer.," in *CVPR*, IEEE, 2017.

[52] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz, "A closed-form solution to photorealistic image stylization.," in *ECCV*, IEEE, 2018.

[53] Z. Shu, S. Hadap, E. Shechtman, K. Sunkavalli, S. Paris, and D. Samaras, "Portrait lighting transfer using a mass transport approach.," *ACM Transactions on Graphics*, 2017.

[54] A. Shashua and T. Riklin-Raviv, "The quotient image: Class-based re-rendering and recognition with varying illuminations.," *PAMI*, 2001.

[55] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity.," *IEEE Transactions on Image Processing*, 2004.

[56] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets.," in *Advances in Neural Information Processing Systems*, 2014.

[57] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 187–194, 1999.

[58] P. Peers, N. Tamura, W. Matusik, and P. Debevec, "Post-production facial performance relighting using reflectance transfer.," *SIGGRAPH*, 2007.

[59] A. Stoschek, "Image-based re-rendering of faces for continuous pose and illumination directions.," in *CVPR*, IEEE, 2000.

[60] Z. Wen, Z. Liu, and T. Huang, "Face Relighting with Radiance Environment Maps.," in *CVPR*, IEEE, 2003.

[61] T. Sun, J. T. Barron, Y.-T. Tsai, Z. Xu, X. Yu, G. Fyffe, C. Rhemann, J. Busch, P. E. Debevec, and R. Ramamoorthi, "Single image portrait relighting.," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 79–1, 2019.

[62] A. Appel, "Some techniques for shading machine renderings of solids," in *Proceedings of the April 30–May 2, 1968, spring joint computer conference*, pp. 37–45, 1968.

[63] A. Georghiades, P. Belhumeur, and D. Kriegman, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose.," *PAMI*, 2001.

[64] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie.," *Image Vision Computing*, 2010.

[65] O. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition.," in *BMVC*, 2015.

[66] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks.," *CoRR*, abs/1812.04948.

[67] P.-W. Grassal, M. Prinzler, T. Leistner, C. Rother, M. Nießner, and J. Thies, "Neural head

avatars from monocular rgb videos," *arXiv preprint arXiv:2112.01554*, 2021.

[68] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European conference on computer vision*, pp. 405–421, Springer, 2020.

[69] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. Guibas, J. Tremblay, S. Khamis, *et al.*, "Efficient geometry-aware 3d generative adversarial networks," *arXiv preprint arXiv:2112.07945*, 2021.

[70] G. Gafni, J. Thies, M. Zollhofer, and M. Nießner, "Dynamic neural radiance fields for monocular 4d facial avatar reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8649–8658, 2021.

[71] S. Athar, Z. Shu, and D. Samaras, "Flame-in-nerf: Neural control of radiance fields for free view face animation," *arXiv preprint arXiv:2108.04913*, 2021.

[72] S. Peng, Y. Zhang, Y. Xu, Q. Wang, Q. Shuai, H. Bao, and X. Zhou, "Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9054–9063, 2021.

[73] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 43–54, 1996.

[74] A. Davis, M. Levoy, and F. Durand, "Unstructured light fields," in *Computer Graphics Forum*, vol. 31, pp. 305–314, Wiley Online Library, 2012.

[75] N. K. Kalantari, T.-C. Wang, and R. Ramamoorthi, "Learning-based view synthesis for light field cameras," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, pp. 1–10, 2016.

[76] E. Penner and L. Zhang, "Soft 3d reconstruction for view synthesis," vol. 36, no. 6, 2017.

[77] M. Waechter, N. Moehrle, and M. Goesele, "Let there be color! large-scale texturing of 3d reconstructions," in *European conference on computer vision*, pp. 836–850, Springer, 2014.

[78] J. Flynn, M. Broxton, P. Debevec, M. DuVall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker, "Deepview: View synthesis with learned gradient descent," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2367–2376, 2019.

[79] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, "Stereo magnification: Learning view synthesis using multiplane images," *arXiv preprint arXiv:1805.09817*, 2018.

[80] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, "Deepvoxels:

Learning persistent 3d feature embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2437–2446, 2019.

[81] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: Learning dynamic renderable volumes from images," *arXiv preprint arXiv:1906.07751*, 2019.

[82] Z. Li, S. Niklaus, N. Snavely, and O. Wang, "Neural scene flow fields for space-time view synthesis of dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6498–6508, 2021.

[83] W. Xian, J.-B. Huang, J. Kopf, and C. Kim, "Space-time neural irradiance fields for free-viewpoint video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9421–9431, 2021.

[84] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-nerf: Neural radiance fields for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10318–10327, 2021.

[85] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5865–5874, 2021.

[86] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.

[87] Y. Feng, H. Feng, M. J. Black, and T. Bolkart, "Learning an animatable detailed 3d face model from in-the-wild images," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–13, 2021.

[88] Z. Shu, M. Sahasrabudhe, R. A. Guler, D. Samaras, N. Paragios, and I. Kokkinos, "Deforming autoencoders: Unsupervised disentangling of shape and appearance," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 650–665, 2018.

[89] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zollhofer, and C. Theobalt, "Stylerig: Rigging stylegan for 3d control over portrait images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6142–6151, 2020.

[90] Y. Zheng, V. F. Abrevaya, M. C. Bühler, X. Chen, M. J. Black, and O. Hilliges, "Im avatar: Implicit morphable head avatars from videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13545–13555, 2022.

[91] C. Wu, D. Bradley, M. Gross, and T. Beeler, "An anatomically-constrained local deformation model for monocular face capture," *ACM transactions on graphics (TOG)*, vol. 35, no. 4,

pp. 1–12, 2016.

[92] K. Kania, K. M. Yi, M. Kowalski, T. Trzciński, and A. Tagliasacchi, "Conerf: Controllable neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18623–18632, 2022.

[93] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, "Learning a model of facial shape and expression from 4d scans.," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 194–1, 2017.

[94] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10529–10538, 2020.

[95] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9224–9232, 2018.

[96] J. T. Kajiya and B. P. Von Herzen, "Ray tracing volume densities," *ACM SIGGRAPH computer graphics*, vol. 18, no. 3, pp. 165–174, 1984.

[97] P. Li, Z. Zhang, Y. Tong, B. M. Foda, and B. Day, "Ilee: Algorithms and toolbox for unguided and accurate quantitative analysis of cytoskeletal images," *Journal of Cell Biology*, vol. 222, no. 2, p. e202203024, 2022.

[98] A. Hou, Z. Zhang, M. Sarkis, N. Bi, Y. Tong, and X. Liu, "Towards high fidelity face relighting with realistic shadows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14719–14728, 2021.

APPENDIX

CONTRIBUTION ON MULTI-AUTHORED PUBLICATIONS

Chapter 2 is based on the paper "ILEE: Algorithms and toolbox for unguided and accurate quantitative analysis of cytoskeletal images" [97]. As co-first-authors, Pai Li and I contributed equally to this work. As an expert in biology, Pai Li was responsible for capturing data samples from plant cells using microscopes. He also collected data samples from other biology labs to ensure the diversity of our dataset. He also utilized his biostatistics background to propose the model for the global gradient threshold. I was responsible for building the linear system and implementing the computational work. We proposed the quantitative indices for the cytoskeleton together. I also built a python library for our pipeline, making our work an easy-to-use tool for biologists. We wrote our part of the documentation respectively. Chapter 3 is based on the paper "Towards High Fidelity Face Relighting with Realistic Shadows" [98]. I contributed to the geometry-related part of the work in this whole research project. I utilized the ray-tracing algorithm to generate shadow masks for face shapes. I also processed all lighting data to provide consistent lighting input based on Spherical Harmonics for the whole pipeline. Andrew Hou proposed the hourglass network structure and designed the loss function. We wrote the corresponding manuscript respectively. Chapter 4 is written solely for the thesis, and may be submitted in the future for publication in vision or graphics venues. I was the only student in this project.