Inferring Differential Structure from Defocused Images

Fahim Mannan

Abstract

In this report, we consider the problem of estimating depth, normal, principal curvatures and their directions at every pixel from two defocused images. Existing literature on depth from defocus (DFD) only considers the problem of depth estimation. On the other hand, surface inference methods usually work with volumetric or point-cloud data. In this project we explore how these two independent techniques perform when they are combined together. Since depth from defocus works with single perspective images, we ensure that single perspective visibility property is maintained throughout the process. Our basic approach is to estimate depth using standard depth from defocus methods and then fit a quadric at every pixel, and finally iteratively refine the local quadrics to infer scene surface properties at every pixel in the image. We took Sander and Zucker's approach [1] and modified it for our problem. The proposed approach is evaluated quantitatively using synthetic data and qualitatively using real defocused images.

I. Introduction

In this report, we look at the problem of estimating differential structure namely - depth, surface normal, principal curvatures and their directions - at every pixel from two defocused images. The two defocused images are images of the same scene but optically blurred by different amount due to different camera parameters. When pictures are taken by finite aperture lenses[1], only a plane in the scene can be in focus. The distance of this plane from the camera's center of projection depends on camera parameters such as the sensor to lens distance and the focal length of the lens. This distance is often referred to as the *focus distance*. The amount of blur at a pixel depends on the focus distance, the size of the aperture[2], and how far the corresponding scene point is from the focal plane. Given the focus distance and the amount of blur at a pixel, we can compute the distance of the corresponding scene point. For a single blurred image, there is an issue of depth ambiguity which is the ambiguity in determining whether the scene point is in front or behind the focal plane. This ambiguity can still be present when a pair of differently defocused images are used. However, this ambiguity can be partially or fully avoided by choosing the camera parameters in a certain way. In this report, we assume that such ambiguity does not exist. There is also a related technique known as Depth from Focus or DFF, that relies on taking a sequence of images with different focus and estimating the depth based on the sharpness of a pixel in those sequence of images. In this report, we do not consider this approach.

Besides estimating depth at every pixel, we would like to get a richer reconstruction of the scene by estimating the surface normal, curvatures and principal directions at every pixel. This allows us to know the property of the surface and perform further processing such as shape segmentation or recognition. The problem of depth estimation and surface inference have been addressed independently in the existing literature. However in this work we look at inferring differential structure in the context of DFD.

A. Issues to consider

Many of the methods iteratively refine the local estimates. However in our case, we have a single perspective image and we want to ensure that the reconstructed scene project to the same pixels. We

Fahim Mannan, School of Computer Science, McGill University.

[1]In practice we cannot have infinitesimally small pinhole aperture lenses.

[2]Often represented in terms of the f-number (N) which is the ratio of the focal length (f) and aperture (A)

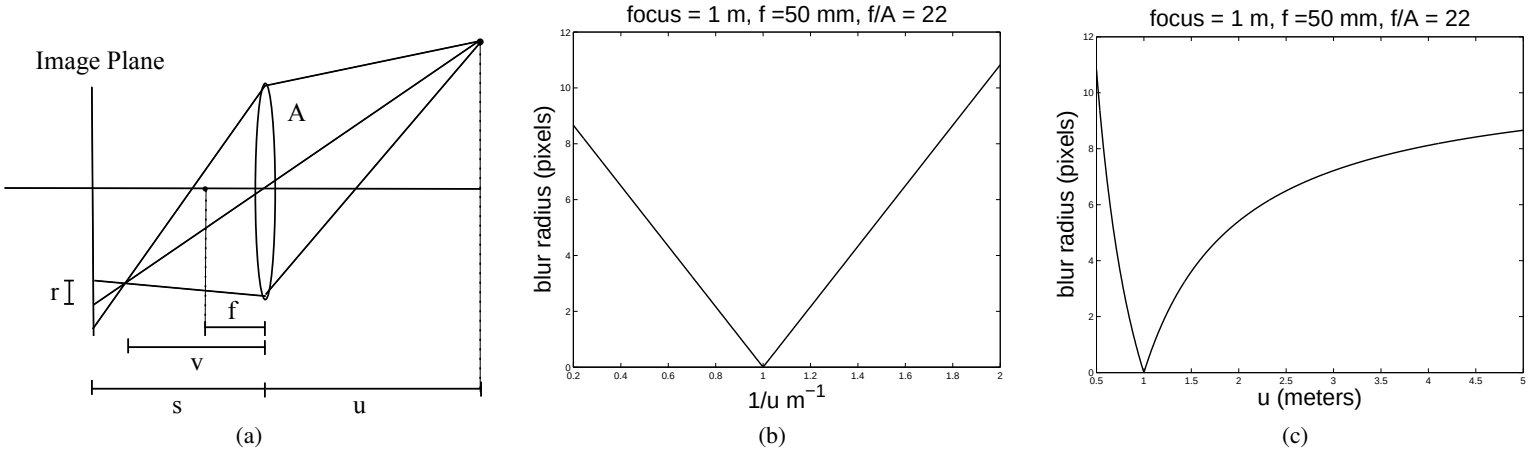


Fig. 1: a) Blur formation process, b) blur radius vs. inverse depth, and c) blur radius vs. depth.

refer to it as the *Reprojection Constraint*. Furthermore most of the surface inference methods work with volumetric or point-cloud data. In our case, we just have two defocused images.

B. Scope

The scope of this project is restricted to exploring the different issues that come up while estimating differential structures from defocused images. Therefore we emphasize more on the differential structure inference stage and on adapting the existing methods for our problem, rather than the initial depth estimation stage.

C. Organization

The organization of this report is as follows. In Sec. II, we give an overview of the underlying concepts used in this work, as well as a review of the relevant literature. Sec. III details the theory and implementation of our work. The results are presented in Sec. IV along with our analysis. Finally, Sec. V concludes this report with a discussion of some of the advantages and disadvantages of our approach along with some possible future research directions.

II. BACKGROUND

This section presents some of the key concepts that are relevant to this work and a review of relevant literature.

A. Depth Estimation from Defocused Images

In the following, we review the main ideas behind estimating depth from a pair of defocused images. We start by looking at the blur formation process, followed by how blurred images can be modeled, and finally how to estimate depth from blurred images. In section III, we provide the implementation details for this project.

a) Blur Formation: Fig. 1a shows how a blurred image can be formed when a scene point off of the focal plane and at distance u from the lens is imaged by a camera with a lens of focal length f and aperture A. The sensor plane is at a distance s from the lens. For a circularly symmetric aperture the shape of the blur is a pillbox. However in practice it is not an exact pillbox, and it is often convenient to model it as a Gaussian of spread σ_G. The relationship between camera parameters and the size of the

blur radius can be expressed by the following equation. In the following equation we express the aperture (A) in terms of the focal length (f) and the f-number (N) i.e. $A = f/N$.

$$\sigma = \rho r = \rho \frac{fs}{2N}(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}) \tag{1}$$

In the above equation, σ denotes the blur radius r in pixels, and ρ is the density of pixels on the sensor plane (i.e. pixels per unit length). It should be noted that the blur can be both positive or negative depending on which side of the focal plane it resides. In terms of blurred image formation, the sign of the blur does not matter for a circularly symmetric aperture. This however causes depth ambiguity, since we do not know if the scene point is in front or behind the focal plane. However in our case, we are using two defocused images, and the camera parameters are chosen in such a way that there is no ambiguity within the operating range of depths. Another point to note is that the blur varies linearly with inverse depth and therefore nonlinear with respect to depth. Figs. 1b and 1c graphically shows this relationship using the absolute value of the blur radius.

b) Blurred Image Modeling: Now that we have seen how a scene point can form a blurred image on the sensor plane, we look at how the blurred image can be mathematically represented. As can be seen from Eq. 1 and the corresponding plots, blur depends on depth. Therefore the amount of blur at each pixel will depend on the distance of the scene point that is projected at that pixel. In other words, the blur in an image varies spatially. However in the case of modeling blurred images it is easier to think of the blur as spatially invariant. Assuming that we are not on a depth edge this requirement is approximately true for most scenes that contain smooth surfaces. With this simplification in mind, the blurred image I, can be represent as the shift invariant convolution of a pinhole image I_0, with a blur kernel H of size σ.

$$I = I_0 * H(|\sigma|) \tag{2}$$

In our problem we have two defocused images and each of them can be represented using Eq. 2.

c) Depth Estimation from Defocused Images: Now we look at how to estimate depth from two defocused images. Depending on how the two defocused images were taken, they can have different amount of blurs for the same scene point. In fact for depth estimation we want them to be as different as possible. Furthermore it is not necessarily that one image is always blurrier than the other. For example if we take two images by varying the focus distance, then the scene points that are closer to the first image's focal plane will be sharper in the first image and blurrier in the second image. And similarly scene points closer to the second focal plane will be blurrier in the first image and sharper in the second image. For this section, let us suppose that the blurrier image is I_B and the sharper image is I_S. Now we want to know how much we need to blur the sharper image to obtain the blurrier image i.e. the relative blur. If we denote the relative blur to be $H_R = H(\sqrt{|\sigma_2^2 - \sigma_1^2|})$, where σ_1 and σ_2 are the blurs in the two images, then we can write the relationship between the blurrier and the sharper images as follows:

$$I_B \approx I_S * H_R \tag{3}$$

Note that this is an approximate model because in practice both the sharper and the blurrier images will contain noise and the above relationship will not be exact. If we assume the noise in the sharper image to be negligible and the blurred image to have zero mean Gaussian noise, then we can formulate the depth estimation problem as the problem of estimating the relative blur between the sharper and the blurrier image as follows.

$$\hat{H}_R = \arg\min_{H_R} \|I_B - I_S * H_R\|^2. \tag{4}$$

If we have a parametric model of the blur kernel then we only need to estimate the blur radius σ_R. For Gaussian kernels it can be shown that the spread of the Gaussian σ_G is related to the pillbox blur radius σ, as $\sigma_G = \sigma/\sqrt{2}$.

B. Local Representation of Surface

In general surfaces can have complex structure and it may not be possible to represent the entire surface in a mathematical form. However we can locally represent a surface using quadrics. In our work we use the Monge representation to locally represent a surface.

$$\phi(u,v) = \{u, v, h(u,v)\} \tag{5}$$

A key requirement of this work is to be able to compute the surface normals and curvatures from a local quadric representation. In the following, we show how this can be accomplished from a local representation of a surface.

a) Surface Normal: The normal can be computed by taking the cross-product of the gradient vectors.

$$\phi_u = (1, 0, h_u) \tag{6}$$

$$\phi_v = (0, 1, h_v) \tag{7}$$

$$N = \frac{\phi_u \times \phi_v}{\|\phi_u \times \phi_v\|} = \frac{(-h_u, -h_v, 1)}{\sqrt{h_u^2 + h_v^2 + 1}} \tag{8}$$

b) Curvatures and Principal Directions: The principal curvatures and their directions are relatively easy to measure at the origin of the quadric from the eigenvalue decomposition of the Hessian at the origin.

$$D^2 h = \begin{pmatrix} \frac{\delta^2 h}{\delta u^2} & \frac{\delta^2 h}{\delta u \delta v} \\ \frac{\delta^2 h}{\delta v \delta u} & \frac{\delta^2 h}{\delta v^2} \end{pmatrix} \tag{9}$$

The eigenvalues correspond to the principal curvatures and the eigenvectors to the principal curvature directions. Computing these quantities away from the origin is a bit more involved and require using the two fundamentals forms from differential geometry, which are known as the first fundamental form or the I form, and the second fundamental form or the II form. The I form relates to the inner product of vectors in the tangent space and the II form corresponds to the local deviation of a surface from its tangent plane. The two fundamental forms are as follows:

$$\mathcal{F}_I = \begin{pmatrix} E & F \\ F & G \end{pmatrix}, \quad \mathcal{F}_{II} = \begin{pmatrix} L & M \\ M & N \end{pmatrix} \tag{10}$$

$$\text{where,} \quad E = 1 + h_u^2, \quad F = h_u h_v, \quad G = 1 + h_v^2$$

$$L = \frac{h_{uu}}{\sqrt{h_u^2 + h_v^2 + 1}}, \quad M = \frac{h_{uv}}{\sqrt{h_u^2 + h_v^2 + 1}}, \quad N = \frac{h_{vv}}{\sqrt{h_u^2 + h_v^2 + 1}}$$

It can be shown that the principal curvatures and their directions are the eigenvalues and eigenvectors of $\mathcal{F}_I^{-1}\mathcal{F}_{II}$ (cf. Def 6.1 of [2]).

$$\mathcal{F}_I^{-1}\mathcal{F}_{II} = \frac{1}{EG - F^2} \begin{pmatrix} MF - LG & NF - MG \\ LF - ME & MF - NE \end{pmatrix} \tag{11}$$

C. Related Works

The idea of estimating depth from defocused images was first proposed by Pentland [3] in the late 80s. In the original work, he proposed two approaches. The first was estimating defocus blur from a single image of a blurred step edge. The second was estimating defocus blur from a pair of sharp and blurred images. In practice we cannot have a true sharp image. Therefore later methods dealt with two defocused images of the same scene taken with two different camera parameters. Many different methods have been proposed ranging from frequency domain methods such as [4], to spatial domain methods e.g. [5], [6]. Our method falls into the category of spatial domain based relative blur based approaches. One of the main works in this area is by Ens and Lawrence [6], who estimates the relative blur kernel from two

defocused images in a training phase and later uses that to estimate depth for arbitrary images taken with the same camera settings. More recent works that uses relative blur and assumes the PSF to be Gaussian are by Favaro et al. [7]–[11]. In these works the authors propose inhomogeneous isotropic diffusion based approach where the diffusion coefficient varies at every pixels and corresponds to the relative blur at those pixels. They formulate the problem as an energy functional consisting of a data term similar to Eq. 4 and regularization terms to obtain a smooth depthmap. The energy functional is optimized using variational methods.

Most DFD methods do not perform any additional inference such as inferring the shape of a surface. Surface shape is often modeled implicitly as a first order smoothness prior in the variational methods mentioned above. On the other hand there are works that infer 3D surface shape and their properties from noisy data e.g. [1], [12]–[16]. These algorithms usually use 3D range data or volumetric images as input to their method. Sander and Zucker in [1], uses volumetric images to infer the surface trace points along with the normals, principal curvatures and their directions over an estimated smooth surface. The general approach in their work is to first estimate the normal at every voxel using a 3D gradient operator. Then they fit a quadric at every voxel using the estimated normals as constraints for the fit. Finally they use an iterative refinement process to update the normals and curvatures so that neighboring estimates are consistent with each other. Ferri et al. [12] applies this idea to 3D point-clouds obtained from a range scanner. Mathur et al. in [13], proposes to weight the contribution of the neighboring pixels based on how close their previous estimates were compared to the other pixels in the neighborhood. Such adaptive weighting can filter out outliers and result in robust edge preserving estimates. Savadjiev et al. in [14], proposes to build on these works and have a final stage where the surface is recomputed using the estimated surface properties. Fua and Sander in [15] reconstructs surfaces from unstructured 3D points obtained using multiview stereo. Instead of fitting a quadric and then refining the estimates in a separate stage, the authors perform the quadric fitting in an iterative process, each time using the previous estimate to reduce the residual error. Similar to Mathur et al. [13], the authors also weight the contribution of neighboring points based on a distance measure between two quadrics. This is combined with resampling, clustering and triangulation step with visibility constraint to obtain a robust reconstruction.

This work is based on the paper by Sander and Zucker [1]. In most of the papers cited above (except Fua and Sander [15]), the estimation is done in the local coordinate system defined by the tangent to the surface at a specific point. The advantage of doing this the estimation is more stable in the presence of high gradient. Furthermore Sander and Zucker uses an estimate of the normal to determine this local frame. Ferri et al. [12] uses plane fitting in the local neighborhood to estimate the normal. However we found that for this we would need to choose the neighborhood carefully. Because if the neighborhood is too large the fitted plane may be influenced by too many non-local points and the estimate will be off. In our work we do not have any initial normal estimates, and we found that using a quadric fit with a few more parameters give a better initial estimate. We also choose the world coordinate to be the perspective projection space and directly estimate the quantities in this space. This allows us to satisfy the single perspective visibility constraint without using any additional steps or constraints like Fua and Sander. To account for these changes we also derived the necessary equations for normal and curvature estimates from first principles.

III. Depth Estimation and Differential Structure Inference

The algorithmic and implementation details of estimating depth and differential structures from defocused images are described in this section. We first give an overview of the whole process followed by depth estimation using defocused images, surface normal and curvature estimation and their iterative refinement approach.

We can model the problem as optimizing the following energy function.

$$E(Z,N,p_1,\kappa_1,\kappa_2) = E_D(Z) + \sum_{i,j\in\mathcal{N}_i} \Big(\|Z(i)-\hat{Z}(j,i)\|^2 + \|N(i)-\hat{N}(j,i)\|^2 + \|p_1(i)-\hat{p}_1(j,i)\|^2 + \|\kappa_1(i)-\hat{\kappa}_1(j,i)\|^2 + \|\kappa_2(i)-\hat{\kappa}_2(j,i)\|^2 \Big). \quad (12)$$

In the above equation, Z is the depth at every pixel, N the normal, κ_1 and κ_2 are the two principal curvatures and p_1 the direction of the first principal curvature. The first term is the data term that evaluates how well the estimated depth Z models the relative blur between two defocused images. The rest of the terms are smoothness terms and correspond to how consistent the neighboring estimates are. Instead of enforcing the neighboring pixels to have the same estimate, we require that the neighbors of every pixel $\mathcal{N}_i$, is consistent with the estimates of the neighbors. For pixel i and its neighbor j, these are denoted by functions of the form $\hat{f}(j,i)$, where f can be any of the estimated quantities.

A. Overview of the Approach

We optimize objective function in Eq. 12 by first solving for the data term and then using the result for minimizing the smoothness terms. The steps are as follows:

1) Estimate depth from defocused images using the isotropic heat equation to implement Eq. 3 and solving Eq. 4. This will minimize the data term in Eq. 12.
2) Fit initial quadric surfaces at every pixel and estimate the surface normal, curvatures and principal directions using Eqs. 8, 10, and 11.
3) Iteratively refine the above estimates so that they are consistent in a local neighborhood. This step will minimize the smoothness terms in Eq. 12.

In our formulation, we transform the method proposed by Sander and Zucker into a form that is more suitable for our problem. For instance we use a different model for quadric and also perform the operations in a coordinate frame that requires fewer transformations and also satisfies visibility constraint. This is possible because in our problem we have a single perspective view.

B. Depth Estimation

For depth estimation we need to implement Eq. 4. The equation uses the convolution model of blur formation. While we can use Gaussian convolution for this purpose, we decided to use the heat equation. The main reason for this is that, in depth from defocus we need to discriminate small changes in blur and in practice the Gaussian kernel can be evaluated only on a discrete grid. Therefore small changes in blur may not be adequately captured by a discrete Gaussian kernel. By using the heat equation with sufficiently small step size we can simulate small changes in blur.

$$I_t = \Delta I = I_{xx} + I_{yy} \quad (13)$$

$$I_t^n = I_t^{n+1} + \delta t(I_{xx} + I_{yy}) \quad (14)$$

$$\sigma^2 = 2t = 2\delta t N \quad (15)$$

The heat equation and its numerical form are given in Eqs. 13 and 14. For a pair of defocused images with known camera parameters (i.e. focus distance, aperture, etc.) and an estimated range of depths, we first estimate the maximum relative blur. Then we find the number of iteration by choosing an appropriate δt using Eq. 15. The two images are blurred independently using isotropic diffusion. At this point we have two stacks of blurred images $\mathbf{I}_1$ and $\mathbf{I}_2$. Using these two stacks we compute the error term in Eq. 4 and concatenate them together. In pseudo-code:

$$D = \{\|\mathbf{I}_2(N:1) - I_1\|^2,\ \|I_2 - I_1\|^2,\ \|\mathbf{I}_1(1:N) - I_2\|^2\} \quad (16)$$

Here D is a 3D matrix, where each 2D slice is the cost of applying some relative blur. The index in the 3rd dimension corresponds to the relative blur. For instance the index in the middle of the 3rd dimension

corresponds to $\sigma_R = 0$, and indices less than that corresponds to negative or positive relative blur. Note that the sign here is arbitrary and is only used to distinguish between whether the first or the second image is blurred. It is often helpful to slightly blur the data cost matrix to remove noisy estimates. The initial depth map is obtained simply by assigning the index that minimizes the data cost at each pixel and looking up the corresponding depth value of that index.

C. Differential Structure Estimation

For estimating the differential structure we follow the approach proposed by Sander and Zucker in [1]. However in our case we want to update the depth as well and ensure that visible scene points remain visible and project to the same pixel. This constraint that we call the reprojection constraint is explained first. Then we look at the quadric surface fitting and how to estimate surface properties.

a) Reprojection Constraint and Coordinate System: The reprojection constraint ensures that the distance of the scene point from the camera does not affect the pixel that it is projected to. In other words if we reproject the estimated 3D points to the 2D image, they will project to the same pixels as the original input image. In the basic pinhole camera image formation model, a scene point at (X, Y, Z) is projected to the image plane at pixel (x_i, y_i, z_i). If the image plane is at a distance $s_\rho = \rho s$ measured in pixels (recall that ρ is the pixel density) from the center of projection, we have the following relationship.

$$\begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} = \begin{pmatrix} s_\rho & 0 & 0 & 0 \\ 0 & s_\rho & 0 & 0 \\ 0 & 0 & s_\rho & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{17}$$

$$\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} \frac{x}{w} \\ \frac{y}{w} \\ \frac{z}{w} \end{pmatrix} = \begin{pmatrix} \frac{s_\rho X}{Z} \\ \frac{s_\rho Y}{Z} \\ s_\rho \end{pmatrix} \tag{18}$$

From this we can see that as the depth of a scene point Z changes, its projection on the image plane (x_i, y_i) also changes. To ensure that scene points project to the same point even after their distance has changed, we need to change (X, Y) coordinates as well, so that $(X/Z, Y/Z)$ remains constant. This constraint can be ensured if we choose the coordinate system to be in the projection space.

An issue with Eq. 17 is that the perspective projection matrix is not invertible. Fortunately for our problem we have a depth estimate at every pixel. Therefore we have,

$$\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} \frac{s_\rho X}{Z} \\ \frac{s_\rho Y}{Z} \\ \hat{Z} \end{pmatrix} \tag{19}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \hat{Z}/s_\rho & 0 & 0 \\ 0 & \hat{Z}/s_\rho & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \tag{20}$$

Eq. 20 is used to project back the points from image space to 3D space.

b) Quadric Surface Model: The quadric surface model that is used for our surface is of the following form:

$$\begin{aligned} \phi(u,v) &= \{u, v, h(u,v)\} \\ \text{where,}\quad h(u,v) &= \frac{1}{2}au^2 + buv + \frac{1}{2}cv^2 + eu + fv + g \\ &= \frac{1}{2}\begin{pmatrix} u & v & 1 \end{pmatrix}\begin{pmatrix} a & b & e \\ b & c & f \\ e & f & 2g \end{pmatrix}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix} u & v & 1 \end{pmatrix} Q \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \end{aligned} \tag{21}$$

In the original Sander and Zucker's work, they had a volumetric image, and therefore an estimate of the normals at every voxel. This required them to take those normal estimates in the quadric fitting step. After fitting quadrics at every voxel they ended up with quadrics estimated in the local coordinate frame. In our case we do not have any initial normal estimate, and so we need to estimate the normal along with the quadric. Ferri et al. in [12] used plane fitting at every point to estimate the normal. However one problem with that is that the neighborhood needs to be chosen judiciously so as not to throw off the normal estimate by points that are on a different plane than that of the origin. We chose our quadric to be of the form in Eq. 21, so that the normals can be estimated as part of the quadric. Of course this also means that the quadric is no longer estimated locally in the tangent plane coordinate system but in the global coordinate system, which in our case is the perspective projection coordinate.

The normals, principal curvatures and their direction can be computed using the equations given in the previous section. For surface normal at every pixel we have the following equation.

$$N(u,v) = \begin{pmatrix} -a & -b & -e \\ -b & -c & -f \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{22}$$

We now look at how we can obtain a quadric at a neighboring point using the estimated depth Z, normal N, principal curvatures κ_1, κ_2, and their directions p_1, p_2. The quadric at a point can be formed using these estimates as follows:

$$R = \begin{pmatrix} p_1 & p_1^{\perp} \end{pmatrix} \tag{23}$$

$$\bar{n} = \begin{pmatrix} \bar{e} \\ \bar{f} \end{pmatrix} = R^T \begin{pmatrix} -\frac{N_1}{N_3} \\ -\frac{N_2}{N_3} \end{pmatrix} = R^T n \tag{24}$$

$$g = Z \tag{25}$$

$$\begin{aligned} \bar{Q} &= \frac{1}{2}\begin{pmatrix} \kappa_1 & 0 & \bar{e} \\ 0 & \kappa_2 & \bar{f} \\ \bar{e} & \bar{f} & 2g \end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix} K & \bar{n} \\ \bar{n}^T & 2g \end{pmatrix} \end{aligned} \tag{26}$$

$$\begin{aligned} \therefore Q &= \begin{pmatrix} R & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix} \bar{Q} \begin{pmatrix} R^T & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix} RKR^T & n \\ n^T & 2g \end{pmatrix} \end{aligned} \tag{27}$$

We can now use Q from Eq. 27 to estimate the curvatures and principal directions using Eq. 10 and 11.

c) Iterative Refinement: In this step, we first estimate the quantities $\hat{f}(j,i)$, at a pixel i from the point of view of its neighbor j. For this we use Eqs. 22 – 27 from above. Then we minimize the smoothness terms in Eq. 12 as follows. For the scalar terms such as the depth and the curvatures, we compute the mean of the neighboring estimates. For vectors N and p_1, we also compute the means and ensure that the vector has unit length. In the following we show how this is done for the normal vector at pixel i. The working dimension of the vector is denoted by k.

$$N_k(i) = \frac{\sum_{j \in \mathcal{N}_i} N_k(j,i)}{N_c(i)}, \quad k = 1 \ldots 3 \tag{28}$$

$$\text{where,} \quad N_c(i) = \sqrt{\sum_{k=1\ldots3} \left(\sum_{j \in \mathcal{N}_i} N_k(j,i) \right)^2}.$$

IV. Experiments and Results

In this section, we present the results that were obtained from our experiments and analyze their performance. First we show some quantitative results for estimating differential structure using synthetic experiments. Then we give results for estimation from real blurred images.

A. Experiment using synthetic data

Here we consider a simple synthetic shape and evaluate how the differential structure estimation process performs for such scene. The quadric used in this experiment is of the following form.

$$z(x,y) = \frac{1}{2}a(x - x_0)^2 + bxy + \frac{1}{2}c(y - y_0)^2 + f$$

For this particular case, the parameter values are, $a = c = f = 10$,$b = -20$,and $x_0 = y_0 = 10$. The surface was generated over a patch ranging between -49 to 49 in both dimensions. The height of the surface was then scaled to 100, and Gaussian noise of $\sigma_N = 10$ was added to every point independently. Fig. 2a shows the original surface and Fig. 2b the noisy version that is used in the experiment. The corresponding contour plots are shown in Fig. 2c and 2d to illustrate how noise affects the contour shape.

An initial quadric fit was performed on patches of size 21×21 followed by iterative refinement using patches of size 7×7. The result of running iterative relaxation process for this input is shown in Fig. 3. The top row of the figure shows the contour plot along with the 2D projection of the normals. The results are superimposed on the ground-truth obtained by applying quadric fitting on the entire noisy surface. From Fig. 3a it can be seen that the initial fit does a reasonably good job at estimating the general surface structure. However the depth and normal estimates are still noisy. In successive iterations the estimates are more consistent with the ground-truth. Fig. 3d and 3e shows the sum of squared error between the ground-truth and estimates, and the residual error respectively. From the residual error plot it can be seen that the residual goes down in every iteration as expected. However the sum of squared error decreases up to iteration 3 and then slowly starts to increase. This is not so surprising given that the smoothness terms do not take into consideration how well the estimates fit the observed images.

B. Experiment using defocused images

In this section, we present the experiments performed using real defocused images as shown in Fig. 4. Here we first compute the initial depth estimate using the depth estimation method discussed in Sec. III-B. The near and far focal planes are at 529mm and 869mm and the entire scene is contained within these two focal planes. The maximum relative blur raidus was taken to be $\tilde{1}.18$ pixels and the step size of the diffusion equation was $\delta t = 0.025$. This resulted in 28 iterations per image and a total of 57 depth

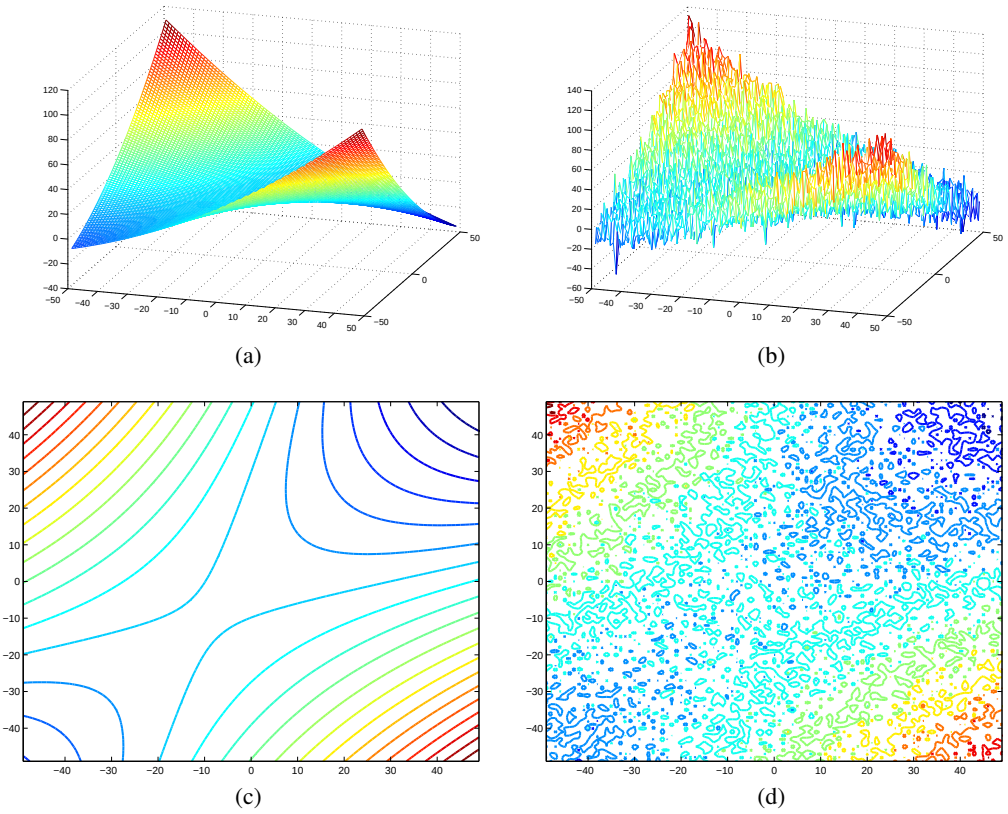
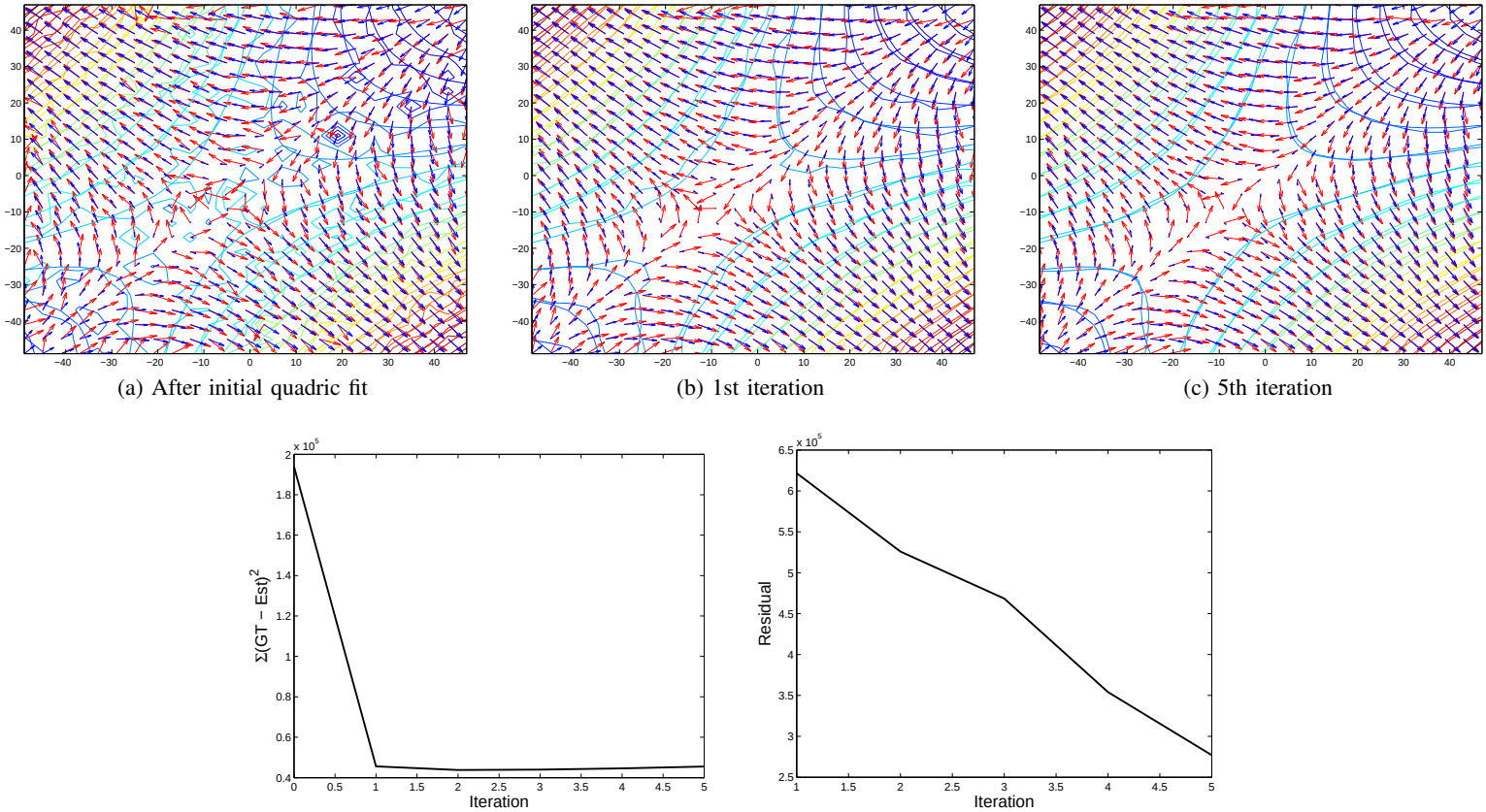


Fig. 2: a) The original surface and b) its noisy version that was used for the synthetic experiment. The bottom figures show the corresponding contour plots.

labels. The data cost is averaged in a local circular neighborhood of diameter 3 pixels. The result of this stage is shown in Fig. 5a. Although for the most part the depthmap is reasonably estimated, there are noise artifacts on the far end of the table and on the background wall. This initial depthmap is used as input for the later stage. Similar to the synthetic experiment we fit quadric surfaces at every pixel with a neighborhood size of $21 \times 21 \times 20$ (i.e. 20mm depth). In the later refinement stage the size of the neighborhood is $7 \times 7 \times 20$. After the initial quadric fit some of the discontinuities on the near part of the table gets removed but the previous artifacts still remain. As the number of iteration progresses the surfaces become smoother. However the discontinuity edges also become softer. This is in fact an unavoidable outcome of such smoothing process since it does not have any discontinuity preserving term other than the neighborhood size.

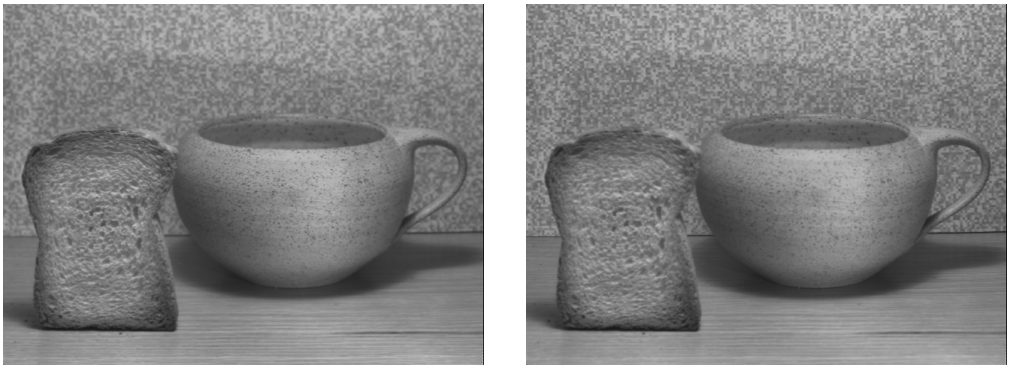
Fig. 6 shows the point-cloud along with the normals obtained by projecting back the 2D depth image into the 3D scene. It should be noted that since the input images are from a single perspective, there will be holes in the scene when viewed from a different perspective. However from the original perspective there should not be any holes (unlike 3D reconstruction from stereo images). The three columns in the figure show the results at different stages of the iteration (increasing from left to right). As the number of iterations increases we can see that there are fewer holes on the table (most of the table surface near the cup is visible in the image). However the smoothness process distributes the points on the edge over the occluded region. Furthermore it can be seen that neighboring normals are somewhat clumped together (note that the direction of the normals affect the shading) which creates an impression of a bumpy surface. Given that the patch size is less than 2% of the entire image this is not too surprising. Larger patch size would have created much smoother surfaces, however it would have also crossed the boundaries of the



(a) After initial quadric fit (b) 1st iteration (c) 5th iteration

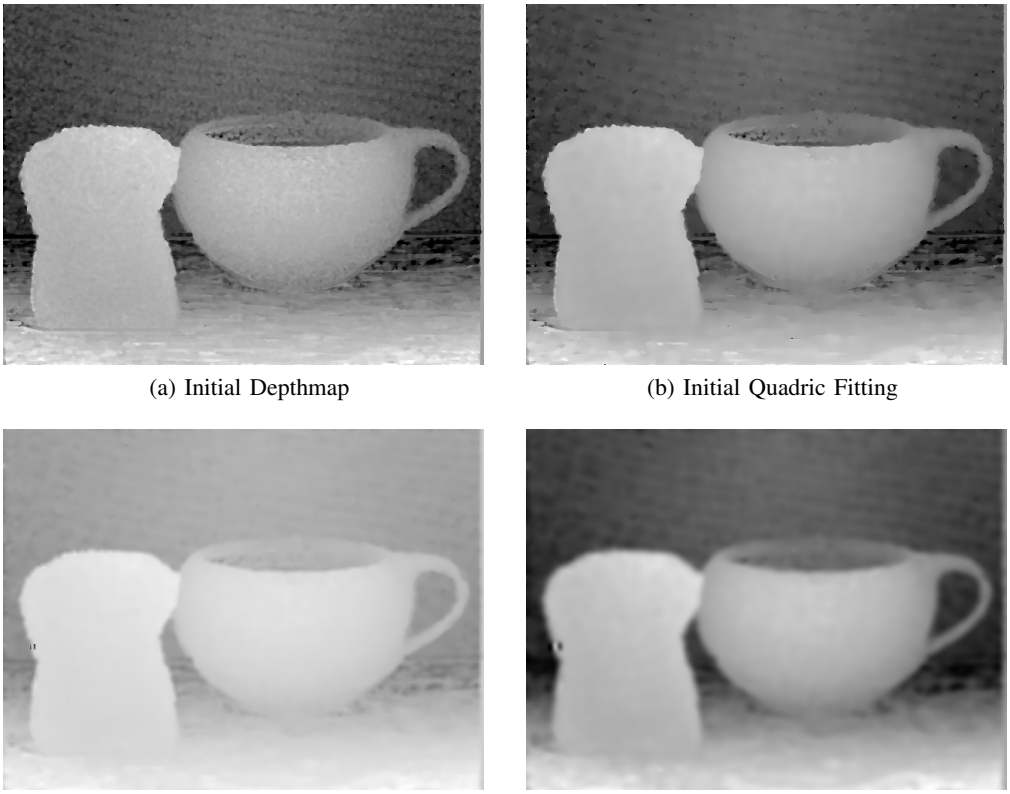
(d) Sum of Squared Error (e) Residual Error

Fig. 3: Top row shows the result of relaxation labeling after different number of iterations. The contours from the estimated surface using the entire noisy data and the surface computed by piece-wise fitting and iterative refinement of quadric are superimposed. The arrows represent the projection of the normal on the 2D plane. The blue arrows correspond to the groundtruth and red arrows correspond to results from piece-wise estimated surface. The estimation error is computed using depth, normal and principal curvatures. The residual error also includes the largest principal curvature direction estimated during the iterative refinement.



(a) Near Focused (b) Far Focused

Fig. 4: Defocused images used for the experiment ("breakfast" scene from [4]). The camera parameters are $f = 25$mm, $N = 8.3$, and $\rho = 39.85$ pixels/mm. The near and far focal points are 529mm and 869mm. The entire scene is contained within these two focal planes. The images are of size 640×480.



(a) Initial Depthmap (b) Initial Quadric Fitting

(c) 1st Iteration (d) 4th Iteration

Fig. 5: Initial depthmap (top-left) and depthmaps from different stages of refinement. The depthmaps get smoother as the number of iterations progresses.

surfaces (e.g. overlapped the cup and the toast) and introduce error in the estimate. The runtime of the algorithm will also increase with patch size. Fig. 7 shows the residual error for the breakfast scene. Similar to the residual error for the synthetic experiment, the residual error decreases in each iteration.

V. Discussion and Future Work

In this report, we have considered the problem of estimating surface properties such as depth, normal, principal curvatures and their direction from two defocused images. For our problem we enforce the reprojection constraint, which ensures that the reconstructed scene points project to the same pixels as in the input defocused images. We proposed an energy formulation model that is a combination of DFD data cost and a modified version of Sander and Zucker's smoothness cost. We optimize the energy function by first optimizing the data cost using Eq. 4. In this step we use discrete depth labels. In the second stage we take the estimated depth map and fit a quadric, followed by iterative refinement of depth, normal, principal curvatures and curvature direction. We perform the quadric estimation in the perspective projection coordinate system, which naturally takes into account the reprojection constraint and also reduces the number of transformations. Finally we showed quantitative results for a synthetic experiment and qualitative results for defocused images.

Some of the advantages of our method are:

1) We do not require any additional visibility constraint. For example, Fua and Sander in [15] needed to enforce visibility constraints in the final triangulation stage of surface reconstruction. By staying in the perspective projection coordinate system we ensure that the scene points corresponding to the pixels remain visible.

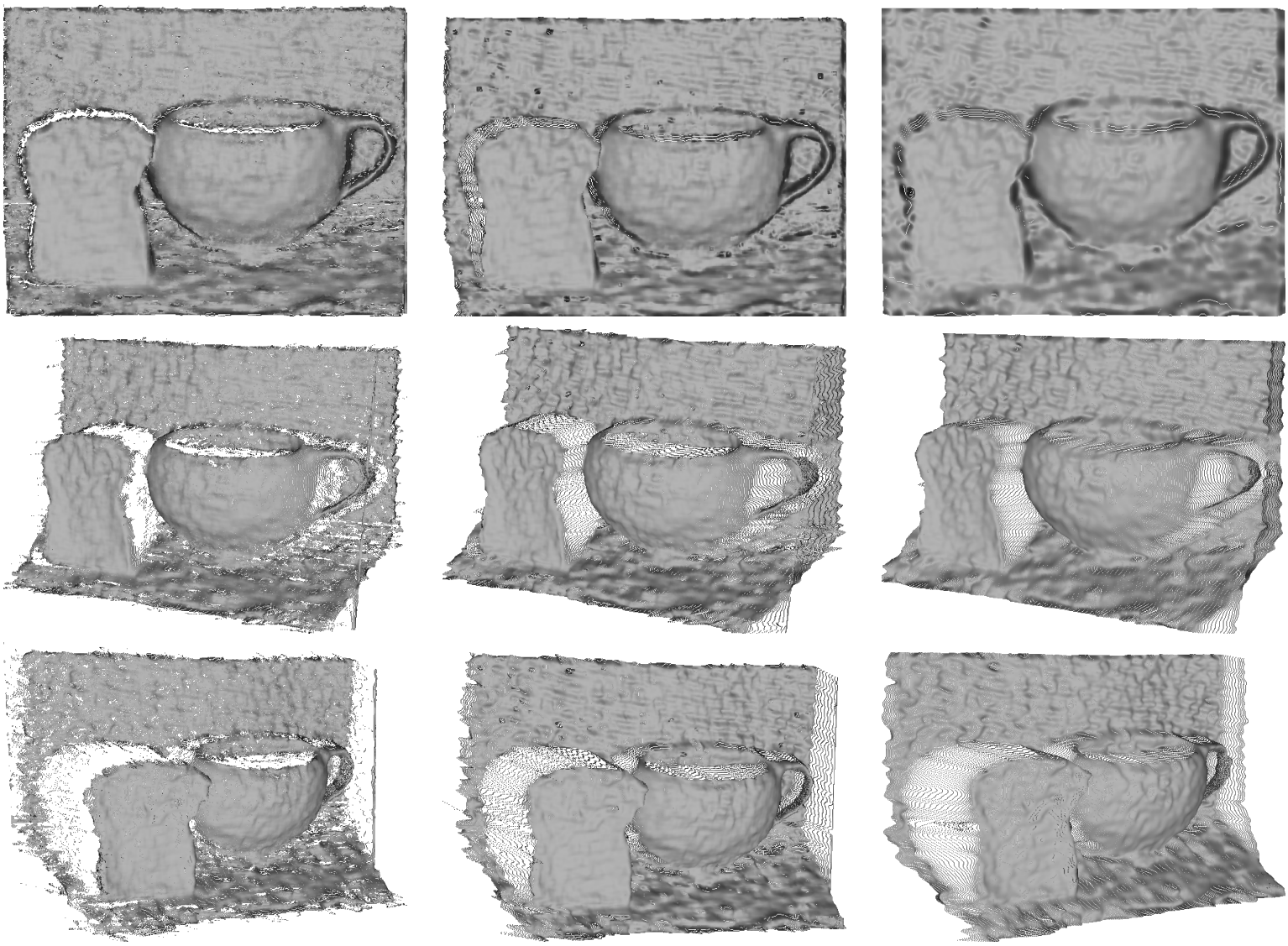


Fig. 6: 3D dense point-cloud visualization from different perspective. The shading is performed using the estimated normal directions at every point. The 1st column corresponds to the initial quadric fitting using 21×21 patches, and the 2nd and 3rd columns correspond to the 1st and 4th iterations using 7×7 patches.

2) By choosing to perform the minimization in the $\mathbb{R}^3$ space, we avoided using additional constraints that Sander and Zucker [1] needed to ensure that the principal direction is orthogonal to the normal. In our case the principal direction is in 2D space and independent of the normal. We just require that it is a unit vector which is similar to the constraint for the normal vector and can be achieved easily.
3) We do not require inverse transformation from the tangent space to $\mathbb{R}^3$. This results in fewer transformations.
4) We start with discrete depth labels. But the iterative refinement step produces smoothly varying depths. However the performance of this final depth map depends on the amount of discretization present in the initial depth map. For our experiment with defocused images we had 57 labels to cover a 34cm region.

Some of the disadvantages of our approach are:

1) We optimize the two terms in Eq. 12 independently. While this simplifies the optimization process,

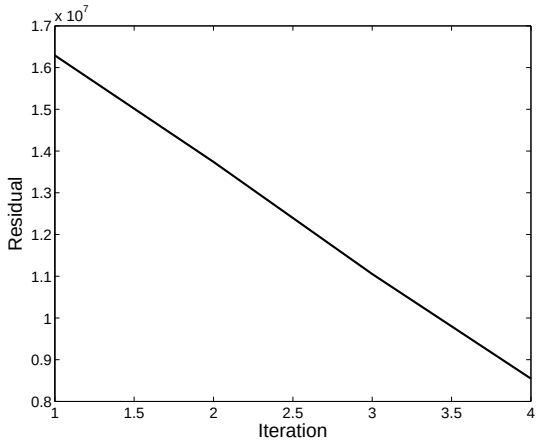


Fig. 7: Residual cost for the breakfast scene. The cost goes down sharply indicating that the neighboring quadrics are becoming more coherent.

it is not a global optimization. Furthermore there is no way to know when to stop the iterative smoothing process. Figures 3d and 3e show that while the residual error can go down, the overall surface can start to deteriorate.

2) The smoothness minimization step assumes that the noise is Gaussian. However this is not true in our case. Noise in depth estimates can be because of weak textures that do not respond well to different amounts of blur. This makes modeling noise for DFD much more difficult.
3) We were able to simplify the estimation process by performing the operations in $\mathbb{R}^3$. While this makes our approach simpler there is also the problem of estimation error near large depth gradients. Our approach will over-smooth such regions.
4) Most of the existing literature deals with single surfaces. However in our case we have multiple surfaces but did not explicitly model this fact. A more robust approach would be to segment the surfaces and perform refinement on the individual surfaces. This would also allow us to use larger patches in the refinement stage and reduce the bumpy artifacts scene in the results.

As for future work:

- The current method is basically smoothing without taking into account the data cost. Further improvement can be done by considering how the data cost is affected while optimizing the smoothness terms.
- Instead of optimizing the data and smoothness terms separately the method can be significantly improved if the two terms are optimized simultaneously within a global optimization framework.
- In the refinement process we use the results from the last iteration. We can use the latest result in the refinement stage to make the residual decrease faster. This is similar to Gauss-Seidel method versus the Jacobi method.
- The initial depth map was obtained using a greedy optimization approach where the depth minimizing the discrete data cost was chosen. This resulted in a discrete depthmap. We can use weighted area and volume flow to perform the initial estimate. In this case, the 3D data cost matrix can be used as the weighting term. The result will be a continuous and smooth depthmap.
- The refined estimates can be used to do a surface reconstruction with the initial data cost and new constraints on the normals and curvatures similar to [14]. In a similar way we can use the refined estimates to segment the surfaces and fit large parts of the surface using a single quadric.
- In Depth from Defocus methods the amount of self occlusion is negligible. We can use multiview defocused images to get occlusion free multiview depthmap which can be merged to obtain a rich 3D reconstruction.

REFERENCES

[1] P. T. Sander and S. W. Zucker, "Inferring surface trace and differential structure from 3-d images," *IEEE PAMITC*, vol. 12, no. 9, pp. 833–854, 1990.

[2] A. Pressley, *Elementary Differential Geometry*, 1st ed. Springer, 2001.

[3] A. P. Pentland, "A new sense for depth of field," *PAMI*, vol. 9, pp. 523–531, July 1987. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.1987.4767940

[4] M. Watanabe and S. K. Nayar, "Rational filters for passive depth from defocus," *IJCV*, vol. 27, pp. 203–225, May 1998.

[5] M. Subbarao and G. Surya, "Depth from defocus: A spatial domain approach," *IJCV*, vol. 13, no. 3, pp. 271–294, 1994.

[6] J. Ens and P. Lawrence, "An investigation of methods for determining depth from focus," *PAMI*, vol. 15, no. 2, pp. 97–108, 1993.

[7] P. Favaro, S. Osher, S. Soatto, and L. A. Vese, "3d shape from anisotropic diffusion," in *CVPR*, 2003, pp. 179–186.

[8] H. Jin and P. Favaro, "A variational approach to shape from defocus," in *ECCV*, ser. ECCV '02. London, UK, UK: Springer-Verlag, 2002, pp. 18–30.

[9] P. Favaro and S. Soatto, *3-D Shape Estimation and Image Restoration - Exploiting Defocus and Motion Blur*. Springer, 2007.

[10] P. Favaro, S. Soatto, M. Burger, and S. Osher, "Shape from defocus via diffusion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 3, pp. 518–531, March 2008.

[11] P. Favaro, "Recovering thin structures via nonlocal-means regularization with application to depth from defocus," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 1133–1140.

[12] F. Ferrie, J. Lagarde, and P. Whaite, "Darboux frames, snakes, and super-quadrics: geometry from the bottom up," *PAMI, IEEE Transactions on*, vol. 15, no. 8, pp. 771–784, Aug 1993.

[13] S. Mathur and F. Ferrie, "Edge localization in surface reconstruction using optimal estimation theory," in *CVPR, 1997.*, Jun 1997, pp. 833–838.

[14] P. Savadjiev, F. Ferrie, and K. Siddiqi, "Surface recovery from 3d point data using a combined parametric and geometric flow approach," in *EMMCVPR*, 2003, vol. 2683, pp. 325–340.

[15] P. Fua and P. Sander, "Reconstructing surfaces from unstructured 3d points," in *In Proc. Image Understanding Workshop*, 1992, pp. 615–625.

[16] G. Guy and G. G. Medioni, "Inference of surfaces, 3d curves, and junctions from sparse, noisy, 3d data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 11, pp. 1265–1277, 1997.