Master Thesis

Real-Time Depth-Image-Based Rendering for Viewpoint-Variable Display on Mobile Devices

Author(s):
Yang, Shuoran

Publication Date:
2016

Permanent Link:
https://doi.org/10.3929/ethz-a-010634330

Rights / License:
In Copyright - Non-Commercial Use Permitted
Real-Time Depth-Image-Based Rendering for Viewpoint-Variable Display on Mobile Devices

Master Thesis CVG Lab
SS 2016

Shuoran Yang

Advisors: Martin Oswald
Professor: Prof. Marc Pollefeys
Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

REAL-TIME DEPTH-IMAGE-BASED RENDERING FOR VIEWPOINT-VARIABLE DISPLAY ON MOBILE DEVICES

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):          First name(s):
YANG  SHUORAN

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the ‘Citation etiquette’ information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date          Signature(s)
ZURICH, 2016.03.09

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.
Abstract

The ability to adjust the display according to the user viewpoint gives a more immersive user experience to augmented reality applications. In this thesis, we propose a pipeline for rendering images to variable viewpoints from a video captured on a mobile device with an arbitrarily aligned camera sensor. We present algorithms to handle depth map restoration, disocclusion and synchronization issues to minimize artifacts in the rendering of virtual images. This pipeline is implemented in the Project Tango framework where an RGB-IR pixel sensor is used to capture depth map and color image inputs. At the end, we demonstrate the results with a video recorded on the Project Tango tablet. The results show that the pipeline is capable of rendering realistic images that have the potential of being used for augmented reality applications.
Acknowledgments

I would like to thank my advisor, Martin Oswald, for his supervision and support throughout my work. This thesis would not have been possible without his guidance and helpful comments. I would also like to thank Thomas Schöps for sharing his knowledge on Project Tango framework and many helpful suggestions. I would like to express my gratitude to Prof. Marc Pollefeys who gave me the chance to work in Computer Vision and Geometry Group. I thank my fellow lab mates in this group for the stimulating discussions. Last but not least, I would like to thank my family for their love and support.
# Contents

Abstract v

Acknowledgments vii

Contents ix

List of Figures xi

List of Tables xiii

1 Introduction 1
   1.1 Project Tango Tablet ............................................. 1
   1.2 Depth-Image-Based Rendering .................................... 2
   1.3 Depth Acquisition ................................................ 3
   1.4 Color-Depth Calibration ........................................... 4

2 Mathematical Foundations 5
   2.1 General Camera Model ............................................. 5
   2.2 3D Warping .......................................................... 6

3 Methodology 9
   3.1 Digital Inpainting Methods ..................................... 9
       3.1.1 Depth Restoration ........................................... 10
       3.1.2 Disocclusion Handling ..................................... 10
   3.2 TV Inpainting Model .............................................. 11
   3.3 Primal-Dual Algorithm ............................................ 11
       3.3.1 Numerical Implementations ................................ 13
   3.4 Image-Driven Regularization ................................... 14
   3.5 Huber Norm for Regularization .................................. 14

4 Implementation 17
   4.1 Depth Map Acquisition .......................................... 17
       4.1.1 Tango Meshing Pipeline ................................... 17
   4.2 DIBR Pipeline ..................................................... 18
   4.3 Synchronization ................................................... 20

5 Experimental Evaluations 23
   5.1 Effects of Depth Acquisition Methods .......................... 23
   5.2 Analysis of DIBR Steps .......................................... 26
   5.3 Evaluation of Depth Map Restoration ........................... 26
   5.4 Different Camera Viewpoints .................................... 26
## Contents

6 Conclusion 33

6.1 Future Work .................................. 33
   6.1.1 Depth Map Acquisition from Multiple Methods .......... 33
   6.1.2 Depth Map Disocclusion with Multiple Sensors .......... 33
   6.1.3 Piece-wise Affine Inpainting .......................... 34
   6.1.4 Head-Tracking with Frontal Camera ...................... 34
List of Figures

1.1 An augmented reality application on mobile tablet that does not match well with the user view. .......................................................... 1
1.2 The front and back of a Project Tango tablet. ................................. 2
1.3 Camera sensors on the back of a Project Tango tablet. ...................... 3
1.4 Depth maps generated from motion stereo and the depth sensor on a Project Tango tablet. (a) and (b) are color and depth images captured with a RGB-IR sensor (c) is a depth map generated by motion stereo algorithm. The grey colored areas in the depth images represent regions that do not have depth information. .......................................................... 4

2.1 Pinhole camera geometry. C is the camera center and p is the principal point. The camera center is located at the original of coordinate frame. figure from [1]. .......................................................... 6

3.1 An example of digital image inpainting. (a) an image with missing pixels (regions painted over with pink strokes) (b) the inpainted image. Figure from [2] 10
3.2 The TV inpainting model estimates the values within inpainting domain $D$ based on the TV norm on extended domain $E\cup D$ and the data constraint on $E$. 11
3.3 Depthmap restoration with uniform and weighted inpainting methods .......... 15
3.4 Comparison of Huber loss and quadratic loss. The green line represents Huber loss at $\alpha = 1$ and the blue line represents quadratic loss. ......................... 15
3.5 Comparison of inpainting results with $l_1$ and Huber norms. (a) represents the input color image and the blue dotted box highlights the region we are comparing. (b) represents the inpainted result of highlighted region using TV model with $l_1$ norm (c) represents the inpainted result of highlighted region using TV model with Huber norm. ............................. 16

4.1 Two depth map acquisition protocols: the flow with solid arrows depicts depth map acquired in a single frame. The flow with dashed arrows depicts an integration of depth map from all previous frames. .................. 18
4.2 Pipeline for taking a pair of depth map and color image to synthesize a depth map and a color image from a different camera viewpoint. .................. 19
List of Figures

4.3 (a) is the input color image (b) is the restored depth map in camera view (c) is the virtual image warped to virtual camera view. The grey areas represent occluded regions in the camera view that become visible in the virtual view. These are the regions that need to be inpainted during disocclusion process. Note that the boundaries that separate foreground objects and the background are divided into the two borders of the occluded regions. (d) is the warped mesh contour generated from the previous 3D warping step. It aligns with the boundaries of the foreground objects in (b). It is used as weight map for inpainting during disocclusion step to give less weight to the foreground boundaries, thus allowing for a depth discontinuities there. 21

5.1 Comparison of input depth maps acquired with different protocols. Column (a) shows the reference image for different frames. Column (b) shows depth maps acquired from the camera sensor in a single frame. Column (c) shows depth maps integrated with depth information from all previous frames. Column (d) shows depth maps with space carving. 24

5.2 Color images acquired from different sources. Column (a) shows the reference image for different frames. Column (b) shows the input depth map is acquired from the camera sensor in a single frame. Column (c) shows the depth map is an integration of depth information from all previous frames. Column (d) shows the depth map is an integration of previous depth information with space carving. 25

5.3 Intermediate steps of DIBR. (a) Reference image from camera sensor (b) Input depth image (c) Inpainted depth image (d) Virtual depth map (e) Virtual color image (f) Inpainted depth map (g) Inpainted color image. 27

5.4 Time distribution among DIBR stages. 28

5.5 Evaluation of TV-inpainting for depth map restoration. (a) Color image at camera pose P (b) Depth map at camera pose P (c) Inpainted depth image of (b) at camera pose P (d) Reference color image at camera pose P' that is two frames before when color image (a) is taken (e) Color image rendered from camera pose P to camera pose P' (f) Difference image between the reference image (e) and the projected image (d). 29

5.6 Evaluation of TV-inpainting for depth map restoration. (a) Color image at camera pose P (b) Depth map at camera pose P (c) Inpainted depth image of (b) at camera pose P (d) Reference color image at camera pose P' that is two frames before when color image (a) is taken (e) Color image rendered from camera pose P to camera pose P' (f) Difference image between the reference image (e) and the projected image (d). 30

5.7 Different Viewpoints. (a) Reference image from camera sensor (b) Virtual depth map at a viewing distance of 15 cm from tablet (c) Virtual color image at a viewing distance of 15 cm from tablet (d) Virtual depth map at a viewing distance of 20 cm from tablet (e) Virtual color image at a viewing distance of 20 cm from tablet. 31
List of Tables
1 Introduction

Mobile devices have become increasingly popular for augmented reality (AR) applications. In current AR applications, the virtual objects are directly overlaying on top of original images captured by the camera. However, depending on the relative poses of user and camera, user view of the scene can be very different from camera view. Therefore, what is displayed on the device can match poorly with what the user should see of the scene. For instance, Figure 1.1 from [3] shows an augmented reality application where the displayed scene is shifted from and disproportionate to the real scene that the user should see. This could cause spatial disorientation for the user. Since the virtual display is dependent only on the camera view, but not on the user view, it also limits the interactivity and flexibility of augmented reality applications.

My work is motivated by this problem of mismatching between user view and camera view that is displayed on mobile devices. Being able to adjust the device display based on the user viewpoint can provide more interactive and immersive augmented reality experiences. In this thesis, we demonstrate a real-time system that is capable of realistic rendering from variable viewpoints on a mobile device with arbitrarily aligned camera sensors.

1.1 Project Tango Tablet

Google’s Project Tango is a platform that enables its devices to learn about their surroundings through depth sensing, motion tracking and 3D reconstruction. As shown in Figure 1.2, a Project Tango tablet has different camera sensors and a front facing screen to display to the user. It is equipped with an RGB-IR camera and a fisheye camera with their layout as shown in Figure 1.3. The RGB and IR images are captured by the same sensor. Therefore, they correspond to the same camera viewpoint. Since the depth map of a scene is computed from

Figure 1.1: An augmented reality application on mobile tablet that does not match well with the user view.
the captured IR image, the depth map and color image will also correspond to the same viewpoint.

A Project Tango device can also track its movement through the scene. It tracks the position and orientation of the user’s device with six degrees of freedom, referred to as its pose. It estimates where a device is relative to where it started using visual-inertial odometry. Standard visual odometry uses camera images to determine changes in position from the relative positions of different features in these images. Project Tango tablets use their fisheye cameras for this purpose since their wide field of view provides more features to track. Visual-inertial odometry supplements visual odometry by adding the capability to track the rotation and acceleration of a device. This allows a Project Tango device to track its orientation and position with greater accuracy.

1.2 Depth-Image-Based Rendering

In depth-image-based rendering (DIBR), a color image and its pre-computed depth map are used to synthesize a new image that mimics what a virtual camera would see from a different viewpoint. When depth information is available for every point in the image, the depth map is paired up with its corresponding color image and the resulting textured-mapped 3D model is used to render novel views.

When a depth map is warped to a novel view, holes and cracks inevitably appear behind the foreground objects. There are two causes for it. Since depth map contains a sampling of depth information over pixels, when the depth map is warped to a virtual viewpoint, it is possible that some pixels on virtual image do not have a corresponding depth value. This problem can usually be handled by an interpolation of the sampled depth values.

Another problem arises when the background areas hidden by a foreground object in the original view needs to be rendered in the synthesized virtual view. This is referred as a problem of disocclusion. Filling out information in the occluded region belongs to the more general problem of digital image inpainting. Since disocclusion often occurs on object boundaries, handling it properly is crucial to the quality of the synthesized virtual image. In Chapter 3, we discusses different image inpainting methods that can be applied to handle disocclusion.
1.3 Depth Acquisition

In DIBR, we need to first attain a depth image for its corresponding color image. Depth acquisition refers to computation of the distance from various points in the scene to a camera. It is one of the most fundamental problems in computer vision and has important applications in 3D reconstruction and image-based-rendering. Many methods have been developed for depth acquisition. These methods generally fall into one of two categories. The first category is based on computer vision techniques such as stereo matching and structure from motion (SFM). These are passive methods which do not rely on active illumination. The second category uses active structured illumination and includes 3D laser scanners, time-of-flight (ToF) cameras and IR depth sensors [4].

Depth sensors on devices such as Microsoft Kinect and Project Tango tablets acquire depth information based on the technology of structured light. They project a pattern of IR light into the scene which rests on the objects as a group of dots invisible to human. An IR camera detects these dots and infers the depth information from deformation of its original pattern.

There are limitations to both categories of depth map acquisition methods. For computer vision based methods, they typically rely on matching of image features from multiple views, which do not work well for non-texture regions. Depth sensors using structured illumination such as IR light patterns typically suffer from limited viewing range and loss of information under certain lighting conditions. For instance, the integrated depth sensor on a Project Tango tablet works best indoors at a distance between 0.5 to 4 meters. Since the depth information is estimated based on viewing infrared light using the device’s camera, areas lit with light sources high in IR, such as sunlight or incandescent bulbs, or objects that do not reflect IR light cannot be scanned well. In general, both categories of methods generate depth maps that have holes.

Figure 1.4 shows depth maps generated from the depth sensor and motion stereo algorithm.
Figure 1.4: Depth maps generated from motion stereo and the depth sensor on a Project Tango tablet. (a) and (b) are color and depth images captured with a RGB-IR sensor (c) is a depth map generated by motion stereo algorithm. The grey colored areas in the depth images represent regions that do not have depth information.

respectively of the same scene on a Project Tango tablet. These holes need to be filled out before the depth map can be used for image-based rendering. In Chapter 3, we discuss different image inpainting techniques that can be applied to restore depth maps.

1.4 Color-Depth Calibration

In DIBR applications, it is crucial that the pair of depth and color images are properly registered. Otherwise, artifacts occur at mismatched regions. Portions of foreground objects could be projected to the background, or part of background will be projected to foreground objects in the rendered views [4].

When the depth and color images are captured by camera sensors, they are generally not synchronized properly. If they are captured by different sensors, they might correspond to two different camera viewpoints. Therefore, a calibration is required to align them properly. If they are taken with the same sensor but at different time, then a change in the scene or movement of the device will cause a mismatch between the two images. In Section 4.3, we explain how the synchronization is achieved in our implementation.
2 Mathematical Foundations

This chapter explains the mathematical formulations in DIBR. Section 2.1 introduces the simplest form of camera model, the pinhole model. Section 2.2 explains how the camera model is applied to reconstruction of a 3D model from depth maps and rendering of images from a new viewpoint.

2.1 General Camera Model

A camera maps between the 3D world and a 2D image. In the pinhole camera model, let the camera be centered at the origin of coordinate system and viewing in the direction of positive z-axis. A 3D point in space with homogeneous coordinate \( \mathbf{X} = (X, Y, Z, 1)^T \) is projected to the image plane at \( z = f \). Its corresponding image point \( \mathbf{x} \), represented by a homogeneous 3-vector \( (x, y, 1)^T \), is the intersection between a line connecting the point \( \mathbf{X} \) to camera center and the image plane, as shown in figure 2.1. The projection can be expressed in matrix form as

\[
\begin{pmatrix}
    x \\
    y \\
    1
\end{pmatrix} =
\begin{pmatrix}
    f & 0 & 0 & 0 \\
    0 & f & 0 & 0 \\
    0 & 0 & 1 & 0
\end{pmatrix}
\begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix}.
\] 

(2.1)

When the principal point is not at the origin of the coordinate frame and the pixel size is not uniform on x- and y- axis, the mapping becomes

\[
\begin{pmatrix}
    x \\
    y \\
    1
\end{pmatrix} =
\begin{pmatrix}
    f_x & 0 & p_x & 0 \\
    0 & f_y & p_y & 0 \\
    0 & 0 & 1 & 0
\end{pmatrix}
\begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix},
\] 

(2.2)

where \((p_x, p_y)^T\) is the offset of principal point on image plane. \( f_x \) and \( f_y \) are focal lengths expressed in pixel units. Equation (2.2) can be reduced to

\[
\mathbf{x} = K[I|0]\mathbf{X}_{\text{cam}}
\] 

(2.3)

where

\[
K = \begin{pmatrix}
    f & 0 & p_x \\
    0 & f & p_y \\
    0 & 0 & 1
\end{pmatrix}.
\] 

(2.4)

The matrix \( K \) is called the camera calibration matrix. \( \mathbf{X}_{\text{cam}} \) emphasizes the 3D point is in the camera coordinate frame. Let \( \mathbf{X} \) be a 3-vector representing the inhomogeneous coordinate
of a 3D point $X$. Then the camera coordinate of a point can be computed from its world coordinate with

$$\tilde{X}_{\text{cam}} = R(\tilde{X} - \tilde{C}),$$

(2.5)

where $\tilde{C}$ represents the inhomogeneous coordinate of the camera center in the world coordinate frame and $R$ is the rotation matrix representing orientation of the camera coordinate frame. Equation 2.6 can be written in matrix form as

$$X_{\text{cam}} = \begin{pmatrix} R & -R\tilde{C} \\ 0 & 1 \end{pmatrix} X.$$

(2.6)

Equation (2.6) can be combined with equation (2.3) to form

$$x = K[R| - R\tilde{C}]X = K[R|t]X,$$

(2.7)

where $R = -R\tilde{C}$. $P = K[R|t]$ is called the projection matrix and (2.7) is reduced to

$$x = PX.$$

(2.8)

### 2.2 3D Warping

Suppose two cameras with different projection centers and poses are viewing the same 3D point in world space and the world origin is at the first camera, which we refer to as the reference camera. Then we have the following matrices

$$P = K[J|0], \quad P' = K'[R|t], \quad C = [0^T, 1]^T,$$

(2.9)

where $P$ and $C$ are the projection matrix and the camera center of the reference camera, respectively. $P'$ is the projection matrix of the second camera. The 3D warping process involves first unprojecting the image point $x$ from reference image into 3D space with

$$\tilde{X} = D(x)K^{-1}x,$$

(2.10)
where $D(x)$ is the depth value at image point $x$ and

$$
X = \begin{pmatrix} \tilde{X} \\ 1 \end{pmatrix}
$$

is the restored 3D point in coordinate frame of the reference camera.

The 3D point $X$ is then projected to the second camera by $x' = P'X$, where $x'$ is the corresponding image point of $x$ on the second image plane. The depth value from the viewpoint of the second camera is the $z$-coordinate of $\tilde{X}' = [R|t]X$. 
3 Methodology

This chapter provides the theoretical background for the inpainting method used for restoring the incomplete depth maps before DIBR and handling disocclusion that occurs during DIBR. Section 3.1 provides an overview of different kinds of inpainting methods and their properties. Section 3.2 explains the general formulation of the TV inpainting method and how it is applied to our work. Section 3.3 presents a primal-dual algorithm for efficient energy optimization in TV inpainting. Section 3.4 and Section 3.5 describe variants of TV inpainting model that improve the inpainting results.

3.1 Digital Inpainting Methods

The goal of inpainting is to fill in missing image information based on the information available. Figure 3.1 shows an example of digital image inpainting. In (a), the image of the scene is incomplete with missing regions covered by the pink strokes. The regions with missing image information are referred to as the inpainting domain and are estimated based on the information from the rest of the image. (b) shows an inpainted image from (a). The objective of inpainting is to fill in the missing regions so it will be non-detectable for an observer who is not familiar with the original image.

Image inpainting has several important applications in computer vision such as digital restoration [5], objects removal [6], or handling disocclusions [7]. But it is a reverse problem that has no well-defined unique solution. All methods operate under the assumption that the unknown regions share the same statistical or geometrical properties with the known parts. They can be divided into three categories based on their prior assumptions [8] as following:

**Diffusion-based inpainting methods** propagate local structure from known to unknown regions based on a smoothness prior. Variants of these methods work well for restoring edges and for filling small regions. However, it tends to give blurry results for large unknown regions;

**Examplar-based methods** are based on self-similar assumption of images. The missing texture is synthesized by sampling from known regions [9];

**Sparse-coding methods** are based on the assumption that the image can be represented sparsely in a given basis, from which the unknown region can be reconstructed.

Due to their different assumptions and characteristics, the examplar-based and sparse-based methods are better than diffusion-based methods at inpainting textured and high-frequency regions [8]. While diffusion-based methods are better for filling small regions and restoring edges.
3.1.1 Depth Restoration

Numerous methods have been proposed to inpaint digital images. However, some models do not perform well on depth maps since they are designed for inpainting textured images. Depth images are non-textured and the unknown regions are generally small compared to the known ones. The depth map is used for image rendering from a new viewpoint, so it is essential that the edge of depth map is well preserved. Otherwise, the foreground will be blended to the background and the rendered image will have many artifacts on its boundary. For this reason, we choose the diffusion-based total variation (TV) inpainting model to inpaint the depth map. The color images can be used to guide the preservation of edges in corresponding depth maps using a weighted TV-inpainting model as introduced in Section 3.4.

3.1.2 Disocclusion Handling

In DIBR, depth and color images are synthesized from a virtual camera viewpoint. One problem that needs to be handled in this process is disocclusion, which takes place when areas that are occluded in an original view become visible in the virtual view.

Numerous inpainting methods exist to fill in the missing information for both depth and textured images. But typically, the occluded region is surrounded by both foreground and background objects in the virtual view since their occurrence is due to significant depth differences between foreground and background. General image inpainting methods do not recognize the differences between foreground and background. They tend to propagate foreground information as well as background information to the occluded region, which leads to significant artifacts.

Similar to depth image distortion, we extract edge information of foreground objects. Then we use it to guide TV inpainting process to preserve their edges. As a result, only the background information will propagate to occluded regions.
3.2 TV Inpainting Model

The TV inpainting model was first introduced by Chan and Shen in [10] as a model for local, non-texture inpaintings, which is closely connected to the TV restoration model of Rudin, Osher, and Fatemi (ROF) in their pioneering work [11]. Let \( \Omega \subset \mathbb{R}^2 \) be the image domain, \( D \subset \Omega \) an open inpainting domain, and \( E \) a fixed closed domain around \( D \), so \( E \subset \Omega \backslash D \), as illustrated in Figure 3.2. Let \( f : \Omega \backslash D \to \mathbb{R} \) be the observed image with missing information.

The assumption is that the signal in region \( E \) is contaminated by homogeneous white noise. The variational model is to find a function \( u : E \cup D \subset \mathbb{R}^2 \to \mathbb{R} \) such that it minimizes the energy functional

\[
E[u] = \int_{E \cup D} \| \nabla u \| \, dx + \frac{\lambda}{2} \int_E |u - f|^2 \, dx,
\]

where \( \lambda \) is a Lagrange multiplier [10]. In this model, the values within the inpainting domain \( D \) are estimated based on the TV norm on the extended domain \( E \cup D \) and the data constraint on \( E \).

In equation 3.1, \( \| \cdot \| \) is the Euclidean (\( l_2 \)) norm in \( \mathbb{R}^2 \), therefore

\[
\int_{E \cup D} \| \nabla u \| \, dx \equiv \int_{E \cup D} \sqrt{\partial_x u^2 + \partial_y u^2} \, dx \, dy,
\]

and we will use this notation for the rest of this thesis. The TV inpainting model seeks a balance between the TV norm on extended domain \( E \cup D \) and the noise constraint on \( E \), hence the inpainting process is robust to noise. Total Variation uses \( l_1 \) norm which penalizes a big change in gradient as much as a sequence of small changes which sum to the same amount. Hence, minimization of equation 3.1 results in piece-wise constant regions, which allows for better edge preservation than using quadratic regularization term \( \int_{\Omega} \| \nabla u \|^2 \, dx \) in the classical Tikhonov regularization approach [12].

We assume all the available depth information in the input depth map is correct. Therefore, the noise constraint term evaluates to 0, and the model is reduced to

\[
\min_u E[u] = \int_{\Omega} \| \nabla u \| \, dx, \quad \text{subject to } u = f \text{ on } \Omega \backslash D. \tag{3.2}
\]

3.3 Primal-Dual Algorithm

The TV norm is hard to optimize due to its non-differentiability at 0. Therefore, methods have been developed that make use of duality theory. By introducing a dual variable \( \omega : \Omega \to \mathbb{R}^2 \),
the non-smooth minimization problem over \( u \) can be transformed into a smooth saddle-point problem in \( u \) and \( \omega \). As a result, simple gradient-based optimization algorithms like steepest descent can be applied.

As proposed in [13], we introduce a dual variable \( \omega : \Omega \to \mathbb{R}^2 \) that satisfies

\[
\|\nabla u\| = \max_{\|\omega\| \leq 1} \nabla u \cdot \omega,
\]

following the Cauchy's inequality. Therefore, equation (3.2) has the equivalent form

\[
\int_{\Omega} \|\nabla u\| dx = \max_{\|\omega\| \leq 1} \int_{\Omega} \nabla u \cdot \omega dx.
\]

We also know from integration by parts,

\[
\int_{\Omega} \nabla u \cdot \omega dx + \int_{\Omega} u \nabla \cdot \omega dx = \int_{\partial \Omega} u \omega \cdot \mathbf{n} ds,
\]

where \( \omega \) is first order continuously differentiable. Since \( \omega = 0 \) on the boundary of inpainting domain \( \Omega \), the right side of equation (3.5) evaluates to 0. Thus, we have the following equation,

\[
\int_{\Omega} \nabla u \cdot \omega dx = \int_{\Omega} -u \nabla \cdot \omega dx.
\]

We can combine equations (3.4) and (3.6) to get the following equivalent forms for energy functional in (3.2),

\[
\int_{\Omega} \|\nabla u\| dx = \max_{\|\omega\| \leq 1} \int_{\Omega} \nabla u \cdot \omega dx = \max_{\|\omega\| \leq 1} \int_{\Omega} -u \nabla \cdot \omega dx.
\]

With this formulation, the TV model becomes a saddle point problem

\[
\min_{u} \max_{\|\omega\| \leq 1} E(u, \omega) = \min_{u} \max_{\|\omega\| \leq 1} \int_{\Omega} \nabla u \cdot \omega dx = \min_{u} \max_{\|\omega\| \leq 1} \int_{\Omega} -u \nabla \cdot \omega dx.
\]

Based on min-max theorem, we can interchange the min and max to obtain

\[
\min_{u} \max_{\|\omega\| \leq 1} \int_{\Omega} -u \nabla \cdot \omega dx = \max_{\|\omega\| \leq 1} \min_{u} \int_{\Omega} -u \nabla \cdot \omega dx.
\]

Hence, the TV model can be represented as

\[
\min_{u} E(u, \omega) = \min_{u} \int_{\Omega} -u \nabla \cdot \omega dx
\]

in primal form and

\[
\max_{\|\omega\| \leq 1} E(u, \omega) = \max_{\|\omega\| \leq 1} \int_{\Omega} \nabla u \cdot \omega dx
\]

in dual form. Notice that in the dual form, the objective functional \( E(u, \omega) \) is differentiable in dual variable \( \omega \). We can solve this with the Primal-Dual algorithm as introduced in [14] with the following iterative steps:
1. **DUAL STEP** Fix $u = u^k$ and apply one step of gradient ascent to the maximization problem. The ascent direction is computed as a partial derivative on $\omega$

$$\frac{\partial}{\partial \omega} E(u^k, \omega) = \nabla u^k$$

and the update is

$$\omega^{k+1} = \frac{\omega^k + \nabla u^k}{\max\{1, \|\omega^k + \nabla u^k\|\}},$$

where the normalization term $\max\{1, \|\omega^k + \nabla u^k\|\}$ is applied to enforce constraint $\|\omega\| \leq 1$.

2. **PRIMAL STEP** Fix $\omega = \omega^{k+1}$, apply one step of gradient descent to the minimization problem. The descent direction is

$$\frac{\partial}{\partial u} E(u, \omega^{k+1}) = -\nabla \cdot \omega^{k+1}$$

and the update is

$$u^{k+1} = u^k + \tau \nabla \cdot \omega^{k+1}.$$

Algorithm 3.1 summarizes the primal-dual algorithm we use. Note that $\bar{u}^{k+1} = 2u^{k+1} - u^k$ is an extrapolation step. As proved in [13], the condition $\tau \sigma L^2 < 1$, in which $L$ is the Lipschitz constant of the gradient operator $\nabla$, needs to be met for the algorithm to converge to saddle point solution.

**Algorithm 3.1 Primal-Dual TV-Inpainting Algorithm**

1: **Initialize:**
   Choose $\tau, \sigma > 0, u^0$ as the input image, $\bar{u}^0 = u^0, \omega^0 = 0$.
2: **loop** until convergence
   $$\omega^{k+1} = \frac{\omega^k + \sigma \nabla u^k}{\max\{1, \|\omega^k + \sigma \nabla u^k\|\}}$$
   $$u^{k+1} = u^k + \tau \nabla \cdot \omega^{k+1}$$
   $$\bar{u}^{k+1} = 2u^{k+1} - u^k$$
3: **end loop**

The primal dual algorithm is easy to implement and can be effectively accelerated on parallel hardware such as graphics processing units (GPUs) [13]. This is especially appealing to augmented reality applications where real-time processing is required.

### 3.3.1 Numerical Implementations

In our implementation, the partial derivatives are defined on discrete 2D grid as following:

$$\partial_x^- u_{i,j} = u(i, j) - u(i - 1, j)$$  \hspace{1cm} (3.12)

$$\partial_y^- u_{i,j} = u(i, j) - u(i, j - 1)$$  \hspace{1cm} (3.13)

$$\partial_x^+ u_{i,j} = u(i + 1, j) - u(i, j)$$  \hspace{1cm} (3.14)

$$\partial_y^+ u_{i,j} = u(i, j + 1) - u(i - 1, j).$$  \hspace{1cm} (3.15)

Forward differences $\partial^+$ are used for computing $\nabla$ and backward differences $\partial^-$ are used for computing the divergence to ensure the discretized form of equation (3.6), $\sum \nabla u \cdot \omega = -\sum u \nabla \cdot \omega$, is satisfied.
3.4 Image-Driven Regularization

The discontinuities in a depth image often correspond to object boundaries. Hence, blurred edges in depth maps often lead to noticeable artifacts on object edges in synthesized images computed from DIBR. The visual quality of augmented reality applications will also suffer from inaccurate depth estimation around object boundaries.

In the TV inpainting model introduced in section 3.2, the strength of smoothness is constant independent of the location in the image domain. However, to preserve sharp edges, intra-region smoothing should be preferred over inter-region smoothing. For an object, the smoothing within it should be encouraged more than the smoothing across its boundary. As proposed by Perona and Malik in [15], a spatially varying diffusion coefficient based on image gradient can be used to encourage intra-region smoothing in preference to inter-region smoothing. Following this reasoning, we introduce a weight coefficient to reduce the regularization power on edges and modify the objective functional in (3.2) as

\[ E[u] = \int_{\Omega} g \cdot |\nabla u| dx, \]

where \( g : \Omega \rightarrow R^+ \) is a weighting function determined locally based on the magnitude of the gradient of the brightness function as following,

\[ g(x) = \frac{1}{1 + \|\nabla u(x)\|^2}. \]

The edges in an image have higher gradient and therefore correspond to a lower weight for its smoothness term. For a depth map, this allows for a sharp change in depth value, corresponding to the boundary of an object. For a color image, this allows for a sharp change in color, also creating the visual effect of an object boundary.

Figure 3.3 shows the results of uniform and weighted inpainting applied to restoration of the same depth map. (a) shows the incomplete depth map captured from depth sensor and (b) shows its corresponding color image. (c) and (d) shows the complete depth map restored from uniform and weighted TV inpainting, respectively. (e) shows the weight map computed from the input color image. It is used as the weight map for weighted TV inpainting. Note that the weight map provides a lower weight for edges, where the pixels are darker, and higher weight for interior of objects, where the pixels are brighter, allowing for depth jumps at edges at a lower cost. It can be seen that depth maps generated with weighted TV inpainting leads to better estimated depth values on object boundaries than uniform TV inpainting, as highlighted in the rectangular box in (c) and (d).

3.5 Huber Norm for Regularization

Using the \( l_1 \) norm for regularization in TV model preserves edges through enforcing piece-wise inpainted regions. But it has the disadvantage of creating stair-casing effects on smooth surfaces that is not parallel to the image plane as shown in image (a) of Figure 3.5. To balance the need of preserving edges and avoiding staircase effects in smooth surfaces, we replace the \( l_1 \) norm with a Huber norm. Hence the energy functional that we aim to minimize becomes

\[ E[u] = \int_{\Omega} \|\nabla u\|_h^2 dx, \]
3.5 Huber Norm for Regularization

![Figure 3.3](image1)

Figure 3.3: Depthmap restoration with uniform and weighted inpainting methods

![Figure 3.4](image2)

Figure 3.4: Comparison of Huber loss and quadratic loss. The green line represents Huber loss at $\alpha = 1$ and the blue line represents quadratic loss.

where $\| \cdot \|_h^\alpha$ is the Huber norm defined as

$$\| x \|_h^\alpha = \begin{cases} 
\frac{|x|^2}{2\alpha} & \text{if } |x| \leq \alpha \\
|x| - \frac{\alpha}{2} & \text{if } |x| > \alpha 
\end{cases} \tag{3.18}$$

The Huber norm enforces smoothed changes when the changes are smaller than $\alpha$ by penalizing it quadratically. It allows for large discontinuity when the changes are greater than $\alpha$ by a linear penalization. Figure 3.5 shows a comparison of Huber and quadratic losses.

With the Huber norm, the TV model becomes

$$\min_u \max_{\|\omega\| \leq 1} \int_{\Omega} \nabla u \cdot \omega - \frac{\alpha}{2} \|\omega\|^2$$

in primal dual formulation [16] and can be solved with primal dual algorithm as discussed in Section 3.3 with slight modifications in the update in dual step as following,
Figure 3.5: Comparison of inpainting results with $l_1$ and Huber norms. (a) represents the input color image and the blue dotted box highlights the region we are comparing. (b) represents the inpainted result of highlighted region using TV model with $l_1$ norm (c) represents the inpainted result of highlighted region using TV model with Huber norm.

\[
\omega^{k+1} = \frac{\omega^k + \sigma \nabla \tilde{u}^k}{1 + \sigma \alpha} \max\{1, \|\omega^k + \sigma \nabla \tilde{u}^k\|\}.
\]  

Figure 3.5 shows the inpainting results using TV model with $l_1$ and Huber norms. It is shown that, with Huber norm, the depth transition is smoother in the more uniform yellow region. But the greater depth changes between the red and yellow regions stays the same for both norms.
4 Implementation

This chapter describes the whole pipeline for synthesizing depth and color images from a virtual view. Section 4.1 discusses different methods for acquiring depth maps and a protocol for attaining the depth map by integrating depth information from all previous frames. Section 4.2 describes how the different steps of DIBR are implemented and linked to each other to render images in a virtual viewpoint. The pipeline is implemented in C++ within the framework of Project Tango. The intrinsics of device’s cameras can be retrieved from the Tango API. The extrinsics can be retrieved between different sensors as frame pairs.

4.1 Depth Map Acquisition

The depth map at a certain time instant can be either computed from a single frame or it can be computed by integrating depth information from all of its previous frames. An integration of depth information is done by extracting a depth map from the output of Tango meshing pipeline which integrates depth maps over time to build a 3D model of the world. Hence, the depth map extracted from it contains depth information from all the previous frames, especially in regions that are difficult to detect due to their angles. The two depth map acquisition methods are demonstrated in figure 4.1.

4.1.1 Tango Meshing Pipeline

The Project Tango pipeline takes a stream of depth maps computed from motion stereo or from depth sensors and incrementally produce a consistent 3D reconstruction by continuous integration of the depth measurements.

To register the depth maps taken from different camera viewpoints, the camera poses corresponding to these depth measurements need to be determined. The pose data of a Project Tango device is estimated from an onboard visual-inertial odometry (VIO) system that fuses data from a wide-angle camera, an inertial measurement unit and 2D feature tracking [17].

Each received depth map is fused into the 3D model by updating a voxel grid called Truncated Signed Distance Field (TSDF) which is used to store the signed distance field of the scene. The TSDF is negative inside an object and positive outside of it. The surface is the zero isocontour of the TSDF. For each depth map, consider arrays shooting from the camera center to the 3D points corresponding to its depth values, which we call end points. For voxels along each array, if it falls within a truncation distance from the end point, the depth value is updated by averaging over all the truncated signed distances stored at this voxel from all depth maps [17].

A depth value not only provides the information that the ray hits a surface at its end point, it also implies the voxels that the ray traverses through before its end point are empty, or only likely to be nonempty within a short distance from the end point, which we represent with the truncation distance. Hence, the voxels along this ray before it hits the end point and
Figure 4.1: Two depth map acquisition protocols: the flow with solid arrows depicts depth map acquired in a single frame. The flow with dashed arrows depicts an integration of depth map from all previous frames.

lying outside of the truncation region will be marked as empty. This in effect carves away any structure stored in between the camera center and end point. This is called space carving in [17] and it improves the surface reconstruction in presence of sensor noise or localization errors.

After the TSDF is updated for a depth map, it is incrementally converted into 3D mesh segments using Marching Cubes, which can be later used for image rendering.

4.2 DIBR Pipeline

Given a pair of depth and color images as input, the DIBR is performed with the following steps: restoration of the incomplete depth map, 3D warping to the virtual view and disocclusion of the warped images. Figure 4.2 provides a graphical illustration of this pipeline.
Figure 4.2: Pipeline for taking a pair of depth map and color image to synthesize a depth map and a color image from a different camera viewpoint.
Depth Map Restoration  In each frame, we take a pair of input depth image and its corresponding color image. Since the depth map is usually incomplete, we perform TV-inpainting of depth values to restore the full depth map. A gradient image is computed from the color image and is used as a weight map for TV-inpainting, such that depth edges are better preserved during the inpainting process. Since the TV inpainting method only uses local information in each iterative step of the energy optimization, it is highly suitable for parallelization and thus is implemented in CUDA.

3D Warping  The restored depth maps and color images are paired up to create a textured 3D model and warped to the new viewpoint. This creates a virtual depth map and a virtual color image. Both images contain holes due to disocclusion. Prior to performing 3D warping, the depth map needs to be preprocessed to form meshes. This is needed to avoid holes on the warped image due to depth sampling. Meshing is implemented using OpenGL for efficiency. Meshes are segmented at object boundaries when the difference in depth values between neighbouring pixels is greater than a threshold, which is 200mm in our implementation. An image of the contour of meshes is generated in this process for the next disocclusion step.

Disocclusion by Inpainting  Since both of the virtual depth map and the virtual color image have holes in occluded regions, we perform a TV-inpainting to fill them out. Since the occluded region is typically on object boundaries, it is important to make sure only the background information is used to inpaint these missing regions. As shown in Figure 4.3, we use the warped mesh contours from previous 3D warping step as the weight map. This allows for a cheaper depth jump across boundaries of foreground objects. So only the background information will be diffused into the missing regions.

4.3 Synchronization

The base code of Project Tango uses Intel Threading Building Blocks (TBB) library for loop parallelism. TBB allows for fast, efficient implementations of dependency graph and data flow algorithms, enabling exploitation of parallelism at a high level of the application. In a flow graph, computations are represented by nodes and communications between channels are represented by edges. When a graph node receives a message, an Intel TBB task is spawned to execute its body object on the incoming message. In our code, depth maps and color images are read from the input channel and are iteratively processed in a separate node. The synthesized virtual depth maps and color images is streamed into another node of the graph to be published to the viewer. It is important that the pairs of depth map and color images are synchronized to the same time stamp to reduce artifacts in rendering.

The RGB-IR sensor records both RGB and infrared images, but the depth image arrives with a delay compared to color image of the same scene due to the time required to be generated from infrared images. When we use the depth map that is obtained from a single frame, as depicted by the solid arrow flow in figure 4.1, the delay is very small. The time stamps for depth map and color image are generally within 0.1s. Therefore, the change in scene is negligible unless the movement is very quick. In our implementation, a depth map captured from a single frame is paired up with the last color image that has already been captured.
4.3 Synchronization

Figure 4.3: (a) is the input color image (b) is the restored depth map in camera view (c) is the virtual image warped to virtual camera view. The grey areas represent occluded regions in the camera view that become visible in the virtual view. These are the regions that need to be inpainted during disocclusion process. Note that the boundaries that separate foreground objects and the background are divided into the two borders of the occluded regions. (d) is the warped mesh contour generated from the previous 3D warping step. It aligns with the boundaries of the foreground objects in (b). It is used as weight map for inpainting during disocclusion step to give less weight to the foreground boundaries, thus allowing for a depth discontinuities there.
When the depth map is extracted from a 3D mesh, as depicted by dashed arrow flow in figure 4.1, there is a greater delay. It takes extra time to update the mesh and to perform a depth map extraction. Therefore, the last received color image has a time stamp that is much later than the last depth map integrated into the mesh. If we extract the depth map by projecting 3D mesh to the camera pose where the color image is taken, there will be a loss of depth information since the depth map corresponding to this camera pose has not been integrated to the mesh yet. For this reason, we store the time stamp of the last depth map integrated to the mesh in our implementation. Then we find the last color image that is received before this time stamp. The camera pose at which this color image is taken is used for extracting the depth map. This color image is then paired up with the extracted depth map for performing DIBR following the pipeline in figure 4.2.
5 Experimental Evaluations

All of the experiments were performed on a desktop with Intel(R) Core(TM) i7 CPU and an NVIDIA Geforce 950 graphics card. In the following sections, we show experimental results that are performed under different settings and protocols.

5.1 Effects of Depth Acquisition Methods

We compare the quality of virtual images rendered from depth maps acquired using different protocols. The experiment runs on a video recorded with the Project Tango tablet. We render depth and color images of size 320 × 180 from a virtual camera that is 15 cm in front of the tablet and looking directly to the center of tablet.

Figure 5.1 compares the depth images acquired from different protocols. The time-integrated depth maps extracted from the incrementally reconstructed 3D mesh contain less depth information for objects at a distance. However, they contain more depth information for close objects, especially on object boundaries, which is very important for the quality of DIBR since inaccurate depth information on object boundaries typically lead to more noticeable artifacts. Comparing columns (c) and (d), it is also shown that space carving leads to less noise in time-integrated depth maps.

Figure 5.2 compares the virtual images rendered with different depth maps at different frames. It shows that by using integrated depth maps, the artifacts are largely reduced compared to the single-frame depth maps, especially on object boundaries such as edges of the table and the sofa. Comparing columns (c) and (d) also shows that space carving on the mesh reduces artifacts on surfaces such as the whiteboard and the radiator.

The runtime of this system depends on many factors, including complexity of the scene, completeness of input images, size of rendered images, etc. In this experiment, the average speed is 2.2 fps for integrated depth map with space carving, 2.3 fps for integrated depth map without carving and 2.8 fps for single-frame depth map. However, this runtime can be much improved when the number of iterations for TV-inpainting is reduced at lower image quality requirement.
Figure 5.1: Comparison of input depth maps acquired with different protocols. Column (a) shows the reference image for different frames. Column (b) shows depth maps acquired from the camera sensor in a single frame. Column (c) shows depth maps integrated with depth information from all previous frames. Column (d) shows integrated depth maps with space carving.
Figure 5.2: Color images acquired from different sources. Column (a) shows the reference image for different frames. Column (b) the input depth map is acquired from the camera sensor in a single frame. Column (c) the depth map is an integration of depth information from all previous frames. Column (d) the depth map is an integration of previous depth information with space carving.
5.2 Analysis of DIBR Steps

Figure 5.3 shows the intermediate steps of DIBR in generating synthesized depth maps and color images. The experimental setting is the same as in section 5.1. We use integrated depth maps with space carving for better depth information. Given a pair of input depth and color images in rows (a) and (b), the depth map is first restored with TV-inpainting with the gradient map of color image as its weight map. Images on row (c) are the inpainted depth maps with the object boundaries well preserved. 3D warping is then performed to render depth and color images to virtual viewpoint as shown in rows (d) and (e). Occluded regions are shown as gray and black regions in depth and color images. They mostly lie on object boundaries as expected, which also indicates that the edges are well estimated in the depth map restoration step. Images on rows (f) and (g) are inpainted virtual depth maps and color images. Despite a few artifacts on object boundaries, the inpainted virtual images are a good representation of what we would expect of the rendered scene. Figure 5.4 represents the time distribution over different stages of DIBR. Restoration of input depth map through TV-inpainting, 3D warping to virtual viewpoint and disocclusion of virtual images through TV-inpainting take 28%, 8% and 64% of the DIBR runtime. Disocclusion takes the longest time because it performs TV-inpainting for depth map and all three channels of color image.

5.3 Evaluation of Depth Map Restoration

In this experiment, we evaluate the accuracy of the TV-inpainting method for restoring input depth maps. We use time-integrated depth maps with space carving as inputs for more accurate depth information. We take pairs of depth and color images taken at camera pose $P$ and render a color image to a previous frame at a camera pose $P'$. The color image taken in this previous frame is used as a reference image. We then compare the reference color image at pose $P'$ with the rendered image by computing their difference image. If the restored depth map is completely accurate, the rendered image should be the same as the reference image, except for in the occluded regions where surfaces are seen from one view but not the other, and the difference image will be completely black. As shown in row (f) of Figure 5.5, the difference image is mostly black, meaning the rendered image is very close to the reference image, with a few pixels’ difference on object boundaries due to non-perfect estimations in depth map inpainting.

Figure 5.6 shows results from another scene. In this scene, the input depth maps are more complete and the baseline is longer than that of the scene in Figure 5.5. Therefore, there are larger non-overlapping regions between the rendered images and the reference images. The overlapping regions in the difference images on row (f) are mostly black, indicating that the rendered images are very close to the reference images.

5.4 Different Camera Viewpoints

In this experiment, we vary the viewpoint by moving the user away from the tablet. We compared the rendered images for the same video input when the user is 15 cm and 20 cm away from the tablet. The results are shown in Figure 5.7. Since we use the tablet as the virtual image plane, when the user is 20 cm in front of the tablet, the effective focal length of
Figure 5.3: Intermediate steps of DIBR. (a) Reference image from camera sensor (b) Input depth image (c) Inpainted depth image (d) Virtual depth map (e) Virtual color image (f) Inpainted depth map (g) Inpainted color image
virtual camera gets longer. Therefore, the rendered images appear closer. There is not much change in the quality of rendered images between two different viewpoints.

**Figure 5.4:** Time distribution among DIBR stages.
Figure 5.5: Evaluation of TV-inpainting for depth map restoration. (a) Color image at camera pose $P$ (b) Depth map at camera pose $P$ (c) Inpainted depth image of (b) at camera pose $P$ (d) Reference color image at camera pose $P'$ that is two frames before when color image (a) is taken (e) Color image rendered from camera pose $P$ to camera pose $P'$ (f) Difference image between the reference image (e) and the projected image (d).
Figure 5.6: Evaluation of TV-inpainting for depth map restoration. (a) Color image at camera pose $P$ (b) Depth map at camera pose $P$ (c) Inpainted depth image of (b) at camera pose $P$ (d) Reference color image at camera pose $P'$ that is two frames before when color image (a) is taken (e) Color image rendered from camera pose $P$ to camera pose $P'$ (f) Difference image between the reference image (e) and the projected image (d).
5.4 Different Camera Viewpoints

Figure 5.7: Different Viewpoints. (a) Reference image from camera sensor (b) Virtual depth map at a viewing distance of 15 cm from tablet (c) Virtual color image at a viewing distance of 15 cm from tablet (d) Virtual depth map at a viewing distance of 20 cm from tablet (e) Virtual color image at a viewing distance of 20 cm from tablet.
6 Conclusion

In this thesis, we developed a pipeline to render virtual images from variable camera viewpoints in real-time on a mobile device. In Chapter 2, we explained the mathematical foundations for DIBR. We gave a description of the projective model for a pinhole camera and how it was applied to DIBR. In Chapter 3, we discussed the general problem of missing image information encountered during depth map acquisition process and DIBR. We discussed different image inpainting methods with an emphasis on their prior assumptions and characteristics. We further introduced an efficient image inpainting technique which restores the depth maps with well-preserved object boundaries. Chapter 4 described how different protocols were used to acquire depth images in Project Tango framework and how the DIBR pipeline was implemented. We also discussed how the critical issue of synchronization is handled in the context of Project Tango framework. Chapter 5 showed experimental results with an analysis of accuracy and efficiency of image rendering under different settings.

6.1 Future Work

There are a few interesting questions that can be further explored.

6.1.1 Depth Map Acquisition from Multiple Methods

Even though our TV-inpainting algorithm can estimate missing depth values to a certain degree of accuracy, the incorrectly estimated depth is still the main cause for artifacts in synthesized virtual images. Due to limitations in both depth sensors and depth computation algorithms, it is difficult to obtain accurate and complete depth maps from either method alone. However, since IR depth sensor works better for flat, non-textured surfaces such as walls, and stereo-based methods work better for scenes with more image features, these two methods can compensate each other to generate a better depth map. An increase in the quality of depth map will certainly lead to more realistic virtual images and better user experiences for AR applications.

6.1.2 Depth Map Disocclusion with Multiple Sensors

When a single camera sensor is used, the occluded regions in virtual view can only be estimated from the geometry of the scene. However, if information of different camera sensors are combined, they can compensate each other for information on the occluded regions. On a mobile device like the Project Tango tablet, the fisheye and RGB cameras are fixed at different locations on the tablet. When the virtual view is rendered from either camera, the occluded regions might be visible from the other camera. In this way, a better virtual image will be rendered combining information from both camera sensors.
6.1.3 Piece-wise Affine Inpainting

The current inpainting method gives a result that is piece-wise constant, but in many cases, the surfaces in the scene are piece-wise affine, which leads to an inaccurate estimation of the depth values. The TV-inpainting algorithm can be modified to be more suitable for estimating piece-wise affine surfaces using methods introduced in [18]. However, this might affect the estimation for some surfaces that do not fit into this assumption. The applicability of this model to different scenarios need to be further explored.

6.1.4 Head-Tracking with Frontal Camera

The frontal camera on a Project Tango device can be used for head-tracking. This can help to determine the user viewpoint and adapt the display accordingly.
Bibliography


