OCCLUSION MANAGEMENT IN CONVENTIONAL AND HEAD-MOUNTED DISPLAY VISUALIZATION THROUGH THE RELAXATION OF THE SINGLE VIEWPOINT/TIMEPOINT CONSTRAINT

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Meng-Lin Wu

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Voicu S. Popescu, Chair
School of Science
Dr. Christoph M. Hoffmann
School of Science
Dr. Daniel G. Aliaga
School of Science
Dr. Xavier M. Tricoche
School of Science

Approved by:

Dr. Voicu S. Popescu
Head of the School Graduate Program

ACKNOWLEDGMENTS

I would like to express gratitude to my advisor, Prof. Voicu Popescu, for guiding me through the journey to become a researcher. This dissertation would not have been possible without his deep knowledge, intense passion, and unwavering support. I would also like to thank professors Daniel Aliaga, Christoph Hoffmann, Elisha Sacks, Xavier Tricoche, and Bedrich Benes for sharing their experiences and insights.

My appreciation goes to fellow graduate students. I am grateful to Jian Cui, Innfarn Yoo, Raine Yeh, Daniel Andersen and Chengyuan Lin for their help and encouragement. I greatly enjoyed working with and learning from them, and I feel honored to have been their colleague. I also appreciate the company of all my friends at Purdue.

I would like to thank my parents for their love. I am forever indebted to them.

Lastly, I would like to thank the Computer Graphics and Visualization Lab Pizza Czar. You know who you are. You are the best.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

ABBREVIATIONS

AR	augmented reality
CGLC	continuous general linear camera
CNN	convolutional neural network
CPU	central processing unit
FPS	frames per second
GPU	graphics processing unit
PPC	planar pinhole camera
PSNR	peak signal-to-noise ratio
RGBD	red, green, blue, and depth; color+depth
ROI	region of interest
VR	virtual reality

ABSTRACT

Wu, Meng-Lin Ph.D., Purdue University, August 2019. Occlusion Management in Conventional and Head-Mounted Display Visualization through the Relaxation of the Single Viewpoint/Timepoint Constraint. Major Professor: Voicu Popescu.

In conventional computer graphics and visualization, images are synthesized following the planar pinhole camera (PPC) model. The PPC approximates physical imaging devices such as cameras and the human eye, which sample the scene with linear rays that originate from a single viewpoint, i.e. the pinhole. In addition, the PPC takes a snapshot of the scene, sampling it at a single instant in time, or timepoint, for each image. Images synthesized with these single viewpoint and single timepoint constraints are familiar to the user, as they emulate images captured with cameras or perceived by the human visual system. However, visualization using the PPC model suffers from the limitation of occlusion, when a region of interest (ROI) is not visible due to obstruction by other data. The conventional solution to the occlusion problem is to rely on the user to change the view interactively to gain line of sight to the scene ROIs. This approach of sequential navigation has the shortcomings of (1) inefficiency, as navigation is wasted when circumventing an occluder does not reveal an ROI, (2) inefficacy, as a moving or a transient ROI can hide or disappear before the user reaches it, or as scene understanding requires visualizing multiple distant ROIs in parallel, and (3) user confusion, as back-and-forth navigation for systematic scene exploration can hinder spatio-temporal awareness.

In this thesis we propose a novel paradigm for handling occlusions in visualization based on generalizing an image to incorporate samples from multiple viewpoints and multiple timepoints. The image generalization is implemented at camera model level, by removing the same timepoint restriction, and by removing the linear ray

restriction, allowing for curved rays that are routed around occluders to reach distant ROIs. The paradigm offers the opportunity to greatly increase the information bandwidth of images, which we have explored in the context of both desktop and head-mounted display visualization, as needed in virtual and augmented reality applications. The challenges of multi-viewpoint multi-timepoint visualization are (1) routing the non-linear rays to find all ROIs or to reach all known ROIs, (2) making the generalized image easy to parse by enforcing spatial and temporal continuity and non-redundancy, (3) rendering the generalized images quickly as required by interactive applications, and (4) developing algorithms and user interfaces for the intuitive navigation of the compound cameras with tens of degrees of freedom. We have addressed these challenges (1) by developing a multiperspective visualization framework based on a hierarchical camera model with PPC and non-PPC leafs, (2) by routing multiple inflection point rays with direction coherence, which enforces visualization continuity, and without intersection, which enforces non-redundancy, (3) by designing our hierarchical camera model to provide closed-form projection, which enables porting generalized image rendering to the traditional and highly-efficient projection followed by rasterization pipeline implemented by graphics hardware, and (4) by devising naturalistic user interfaces based on tracked head-mounted displays that allow deploying and retracting the additional perspectives intuitively and without simulator sickness.

1. INTRODUCTION

In computer graphics and visualization, images are synthesized by sampling the virtual scene with rays that originate from the virtual camera's viewpoint. This virtual camera is modeled after the physical planar pinhole camera (PPC). Specifically, the PPC model assumes that i) sampling is done with rays that resemble light rays, i.e. they are straight lines, ii) all sampling rays converge to a single point, and iii) the rays are defined by a uniform grid of pixels. The PPC model mimics the human eye, and therefore it produces images that are familiar to the user.

The PPC model is well-suited for graphics and visualization, as it generates images that show a virtual scene the same way a photograph or video sequence shows a real world scene. In virtual reality (VR) head-mounted display (HMD) visualization, the familiar visualization using PPC enables the user to feel immersed into the virtual world, and it provides spatial awareness to the user. In augmented reality (AR) HMD visualization, the PPC model is necessary for the virtual annotations to remain perfectly aligned with the real world. Real world scenes are conveniently acquired with video cameras that implement the PPC model, and the resulting image-based datasets support a photorealistic visualization of the scene through simple query and interpolation operations.

However, a region of interest (ROI) can become occluded from the single viewpoint at the current point in time. In image synthesis, occlusions can prevent an ROI from being visible in an image; in real scene acquisition, occluded scene segments are missing from the resulting acquired data set, creating visualization artifacts when the user view exposes them. Occlusion is a fundamental limitation of the PPC camera model, and it cannot be removed no matter how much triangle rendering performance increases, and no matter the sophistication of shading algorithms. Occlusion burdens the user in desktop visualization, as they must manually navigate the virtual scene to

establish line of sight to occluded ROIs. Occlusion is an even more pressing limitation in head-mounted display visualization, where the user must laboriously navigate the scene through physical locomotion.

We propose to reduce the ill effects of occlusion by rethinking the imaging process in graphics and visualization—gainfully deviating from the PPC rendering model. This thesis investigates methods to increase the information bandwidth of images by showing not only what is visible from the current user viewpoint at the current point in time, but also what is visible from additional viewpoints and at additional timepoints. A conventional image is expanded with additional samples while avoiding redundancy and discontinuities. This image generalization is achieved through innovative relaxation of the constraints imposed by the PPC model. The resulting camera model gathers samples across multiple view and time points, infers missing or costly to compute samples, and integrates all samples in an effective visualization. At its core, this thesis advocates a departure from conventional rendering algorithms which aim to reproduce human vision using the PPC model. The thesis advocates expanding image synthesis by exploring the space of camera models that underlie the mapping of the 3D scene to the 2D image. The image synthesis freedom gained through camera model design can lead to powerful and intuitive visualizations of complex datasets. The benefits of image generalization are explored not only in the context of conventional, non-immersive desktop displays, but also in the context of HMDs that provide immersive visualization in VR and AR.

1.1 Motivation

Occlusions are a severe bottleneck for the visualization of large and complex datasets. Conventional images only show dataset elements to which there is a direct line of sight, which significantly limits the information bandwidth of the visualization. Multiperspective visualization (MPV) is a powerful approach for alleviating occlusions to show more than what is visible from a single viewpoint. However,

constructing multiperspective camera models and rendering multiperspective images remains challenging [1].

Immersive navigation in VR and AR mirrors physical locomotion through pose tracking of the HMD. While this navigation modality is intuitive, ROIs in the scene may suffer from occlusion and require significant viewpoint translation. Moreover, limited physical space and user mobility need to be taken into consideration. Some ROIs may require viewpoints that are physically unreachable without less intuitive methods such as walking in-place or redirected walking [2].

Navigation in VR is free from the requirement that virtual and real worlds be perfectly aligned. Therefore, a prevalent navigation technique is teleportation, where the user's virtual position is quickly or instantly translated to the desired destination, while the user is physically stationary. Using teleportation, the user is able to reach previously inaccessible destinations. However, employing teleportation requires the user to first acquire a line of sight to the destination, which can be challenging in complex scenes. Furthermore, the transition of teleportation incurs a loss of spatial awareness by disrupting visualization, and it risks motion sickness by causing the user's physical and virtual positions to diverge [3].

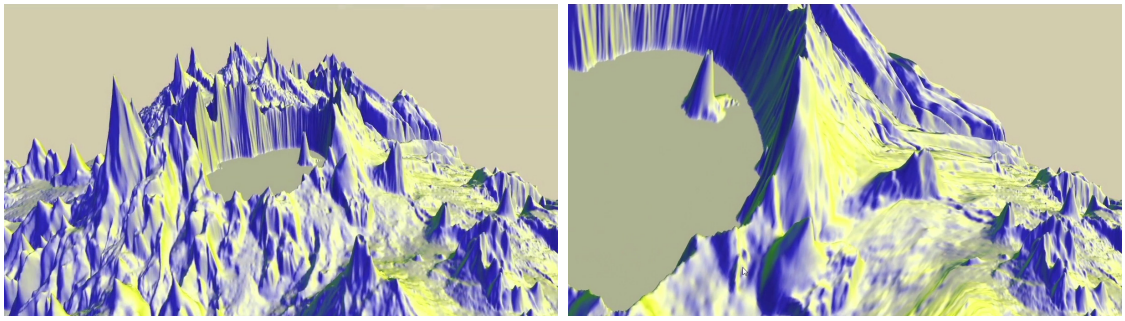
In video acquisition of a real-world object of interest, or target, the acquired video stream can also suffer from occlusion. The target is partially or completely occluded when it moves behind an occluder, or when an occluder moves in front of it. The video inpainting approach removes occlusions from a video stream by cutting out occluders and reconstructing the missing region with samples gathered from other spatiotemporal regions of the video stream. Prior video inpainting methods attempt to optimize the reconstruction according to image quality metrics, but the optimization is in general too computationally expensive for real-time performance. Surveillance and diminished reality applications cannot leverage prior video inpainting methods due to the insufficient performance [4].

1.2 Overview

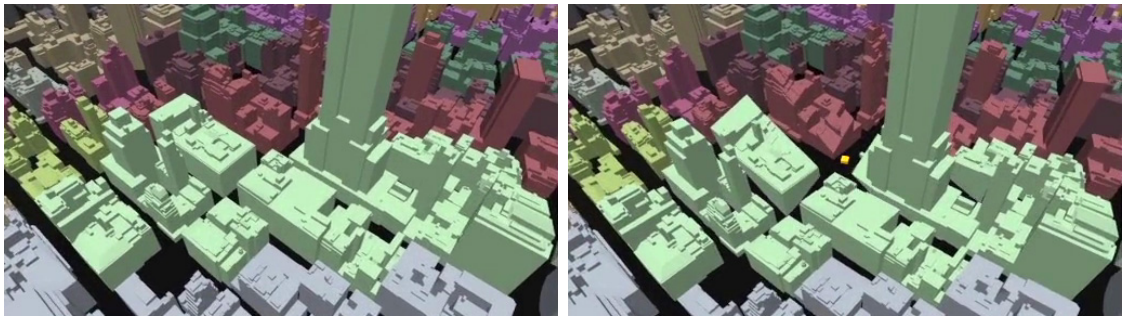
In Chapter 2, we present a framework for designing multiperspective focus+context visualizations with great flexibility by manipulating the underlying camera model. The focus region viewpoint is adapted to alleviate occlusions. The framework supports MPV in three scenarios. In a first scenario, the viewpoint is altered independently for individual image regions to avoid occlusions. In a second scenario, conventional input images are connected into a multiperspective image. In a third scenario, one or several data subsets of interest (i.e., targets) are visualized where they would be seen in the absence of occluders, as the user navigates or the targets move. The multiperspective images are rendered at interactive rates, leveraging the camera models fast projection operation. We demonstrate the framework on terrain, urban, and molecular biology geometric datasets, as well as on volume rendered density datasets (Fig. 1.1).

In Chapter 3, we propose a novel approach for increasing navigation efficiency in VR and AR using MPV. Our approach samples occluded ROIs from additional perspectives, which are integrated seamlessly into the user's perspective. This approach improves navigation efficiency by bringing simultaneously into view multiple ROIs, allowing the user to explore more while moving less. We have conducted a user study that shows that our method brings significant performance improvement in VR and AR environments, on tasks that include tracking, matching, searching, and ambushing objects of interest (Fig. 1.2).

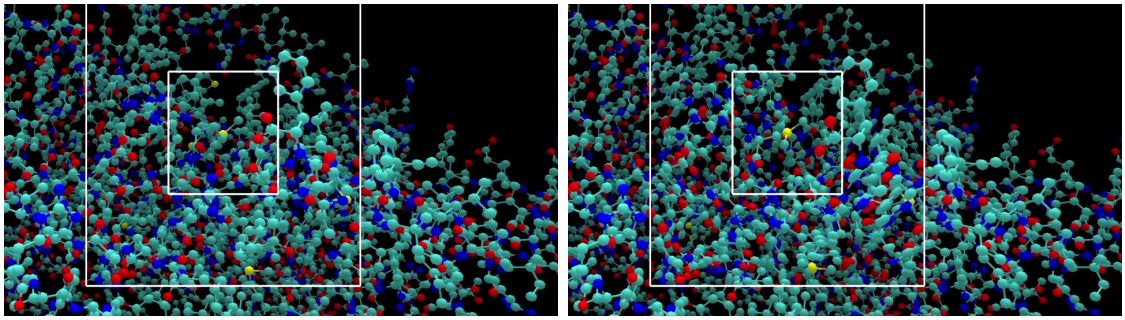
In Chapter 4, we present a novel MPV approach designed to improve navigation efficiency in Virtual Reality applications. The MPV is continuous and non-redundant, it shows the near part of the scene with a conventional, first-person visualization in order to anchor the user, and it is controlled with user head translations and rotations reminiscent of natural navigation. Three types of anchored MPV are introduced, one that provides a lateral disocclusion effect, allowing the user to see around occluders and through side portals (Fig. 1.3, (a)), one that provides a vertical disocclusion



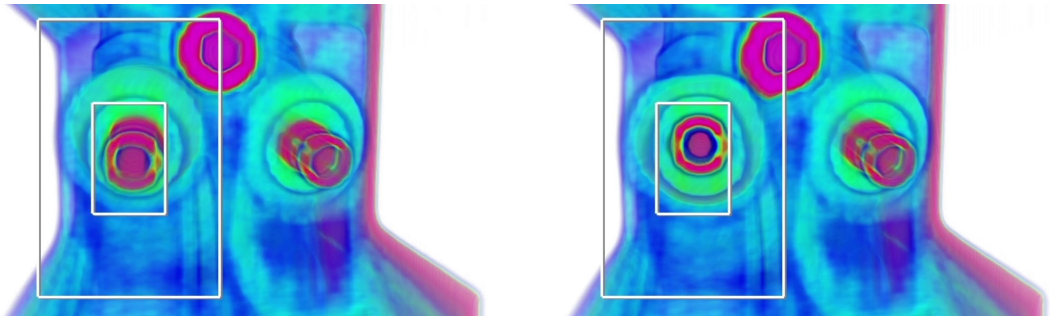
(a) In the PPC image (left), the terrain is seen from a single viewpoint. In the MPV image (right), a sub-frustum freely navigates to focus on the lake.



(b) The ground-level target hidden in the PPC image (left) is revealed in the MPV image (right).

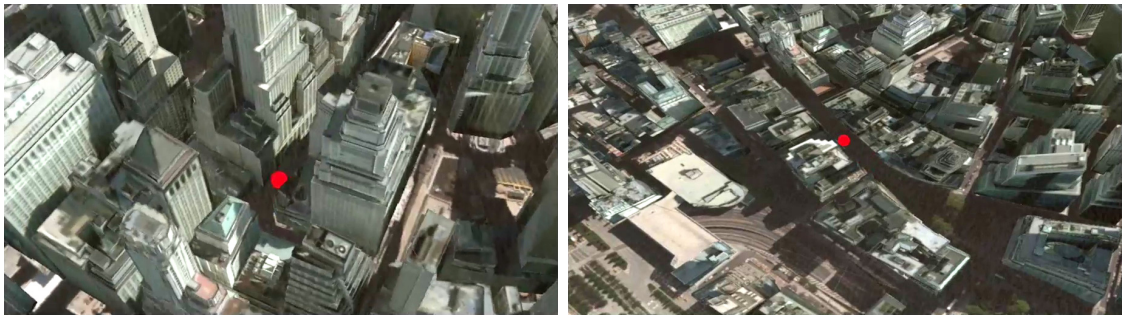


(c) The yellow atom hidden in the PPC image (left) is visible in the MPV image (right).



(d) The left cylinder is viewed from a more favorable viewpoint in the MPV image (right) than the PPC image (left).

Fig. 1.1.: MPV framework demonstrated on various datasets: a) terrain, b) urban, c) molecular biology, and d) volumetric.



(a) Tracking

(b) Matching

(c) Searching

(d) Ambushing

Fig. 1.2.: Multiperspective (right) versus conventional (left) visualization in VR (top half) and AR (bottom half) applications. a) The user tracks a red target that moves on the ground in a VR urban scene. b) The user matches target pairs of the same color pattern in a VR urban scene. c) The user finds a hidden red colored target in an AR scene. d) The user ambushes a pink colored target around the corner in an AR scene.

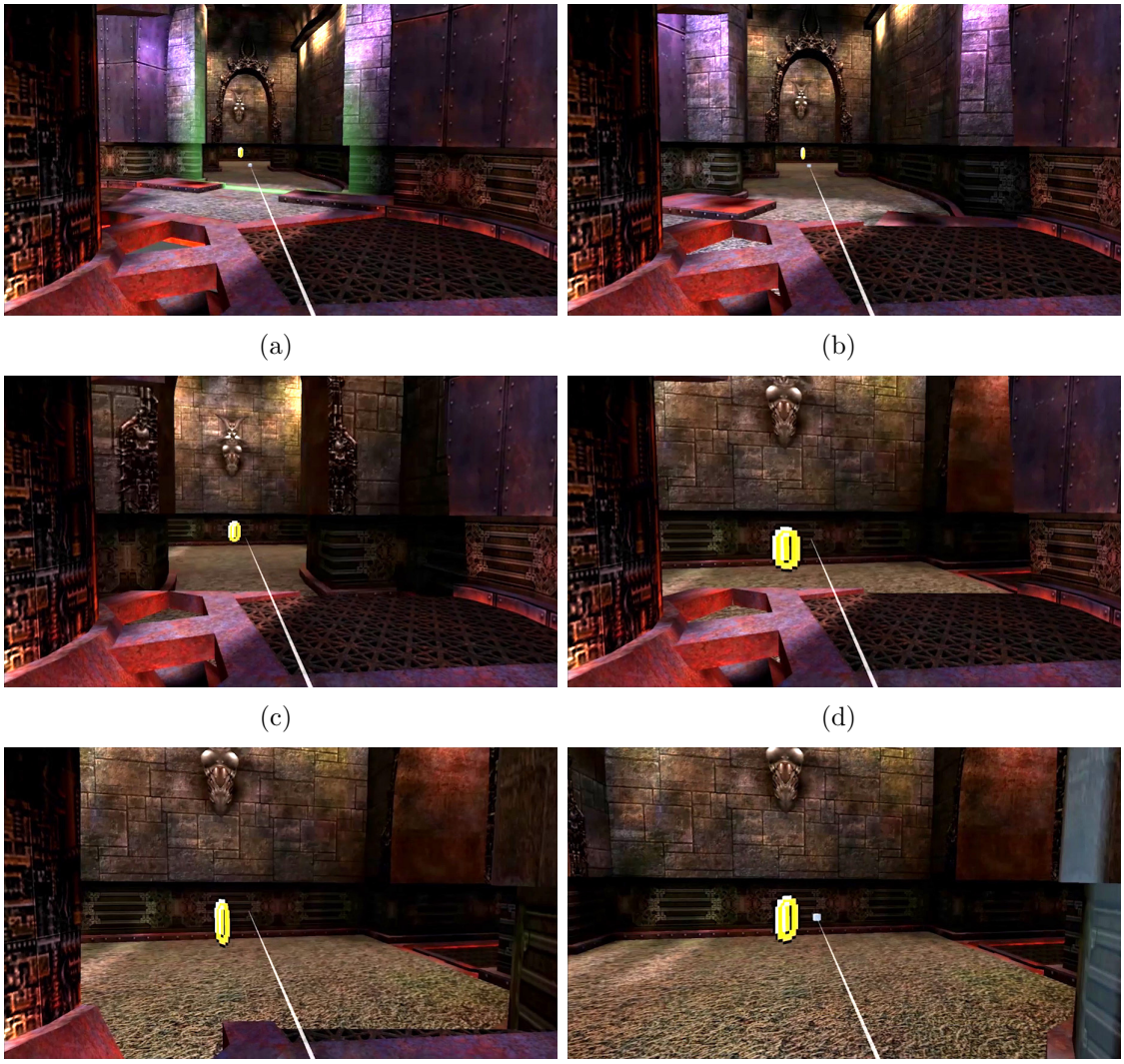
(a) Lateral MPV disocclusion deployed by lateral head translation.

(b) Vertical MPV disocclusion deployed by vertical head translation.

Fig. 1.3.: Naturalistic interface for MPV in VR.

effect, allowing the user to see over and on top of occluders (Fig. 1.3, (b)), and one that provides teleportation, allowing the user to relocate (Fig. 1.4). The VR navigation efficiency benefits of the anchored MPV have been analyzed in a user study. Significant improvements were achieved in the metrics of number of teleportations and total distance traveled. In these metrics, large or greater Cohen's d effect sizes were observed at p-values below 0.05 in a first VR scene, while medium effect sizes at p-values of 0.1 or better were observed in a second VR scene.

In Chapter 5, we present a novel method for fast video occlusion removal to visualize a deformable target. The input to our pipeline is a single RGBD stream acquired from a stationary color+depth camera. Our pipeline segments the current frame to find the target and the occluders, searches for the best matching disoccluded view of the target in an earlier frame, computes a mapping between the target in the current frame and the target in the best matching frame, inpaints the missing pixels of



(a) (b)

(c) (d)

(e) (f)

Fig. 1.4.: Anchored teleportation using MPV. a) The user selects the destination. b) to d) The secondary perspective translates smoothly to the destination, while the primary perspective anchors the user. e) and f) The primary perspective translates to the destination, while the secondary perspective anchors the user.

(a) (b) (c)

Fig. 1.5.: Occlusion in real-world scene visualization (a) removed through multi-timepoint sampling (b), which integrates samples from an earlier frame (c).

the target in the current frame using the mapping to the earlier frame, and visualizes the disoccluded target in the current frame. We demonstrate our method in the case of a walking human occluded by stationary or walking humans (Fig. 1.5). Our method does not rely on a known model of the target or of the occluders, and therefore it generalizes to other shapes. Our method runs at 30fps.

1.3 Thesis Statement

Occlusions in desktop and head-mounted display visualization of 3D datasets can be alleviated by relaxing the single viewpoint and single timepoint constraints of conventional images, to increase the information content of the image by integrating dataset samples collected from multiple viewpoints and multiple timepoints.

2. MULTIPERSPECTIVE FOCUS+CONTEXT VISUALIZATION

2.1 Introduction

Most images used in computer graphics and visualization are computed with the conventional PPC model, which approximates the human eye. Whereas this is essential in applications such as virtual reality where the goal is to make users believe that they are immersed in the scene rendered, researchers in visualization have recognized that the limitations of conventional images are not always warranted. This research path was inspired by artists who have long looked beyond the conventional PPC to achieve compositions that exaggerate scale or avoid occlusions.

One limitation of conventional images is a reduced field of view, which has been addressed with panoramic camera models such as fisheyes. A second limitation is that conventional images sample the dataset uniformly, oblivious to importance variations within the dataset. Conventional focus+context techniques address this limitation by allocating more image pixels to data subsets of higher importance.

A third limitation is that a conventional image samples a dataset from a single viewpoint and occlusions limit the visualization capability of the image. One approach for overcoming occlusions is to rely on the user to change the viewpoint interactively in order to circumvent occluders and to establish a direct line of sight to each data subset of potential interest. One disadvantage of such a sequential exploration is inefficiency: data subsets are explored one at a time, and the navigation path has to be retraced to achieve a systematic exploration of the entire dataset. A second disadvantage is that the user never sees more than a single data subset at a time and connections between subsets that are far apart in the visualization sequence can be missed. The problem is exacerbated in the case of time-varying datasets, where the eloquence of a connection

between distant data subsets could be transient. The problem can be alleviated by visualizing the dataset in parallel with multiple conventional images. The user sees several data subsets simultaneously, but the visualization is discontinuous at the borders of the individual images and the user has to examine one image at the time which reduces the benefits of the parallel visualization.

Another approach for overcoming occlusions is multiperspective visualization, which integrates dataset samples captured from multiple viewpoints into a single, continuous "multiperspective" image. The multiperspective image enables parallel visualization without the high cognitive load of the multitude of disparate contexts presented by individual conventional images [5].

Multiperspective visualization can be seen as a generalization of focus+context visualization. Multiple focus regions are visualized simultaneously connected by continuous context, but without the restriction that all focus regions are visualized from the same viewpoint. The challenge is to construct a multiperspective visualization that provides good control over the multiple viewpoints from where the dataset is sampled, while maintaining image continuity, as needed for visualization efficacy, and while maintaining rendering efficiency, as needed to support interactive visualization and time-varying datasets.

In this chapter we present a flexible framework for interactive multiperspective focus+context visualization. The multiperspective image is constructed by rendering the dataset with a camera with piecewise linear rays. Some of the rays are designed to circumvent occluders and to reach the data subsets of interest, while the remaining rays are designed to connect the data subsets of interest with continuous context. The multiperspective camera is assembled from a small number of *camera segments*, i.e., simple cameras with linear rays, which enables good control over the viewpoints integrated in the visualization. Moreover, the segments provide fast projection which allows rendering the multiperspective image efficiently, on the GPU, by projection followed by rasterization. We also refer the reader to the accompanying video.

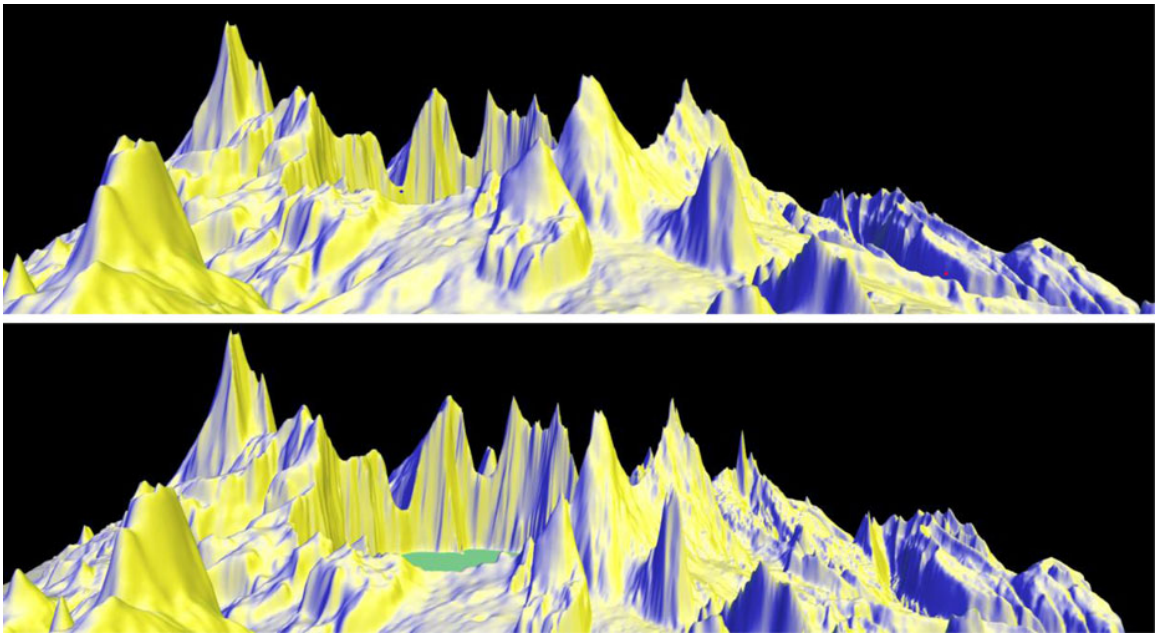


Fig. 2.1.: Conventional visualization of terrain dataset (top) and multiperspective visualization constructed with our framework in top-down fashion (bottom). The viewpoint was modified for two regions individually to reveal a lake (left) and a valley (right).

The framework enables multiperspective visualization according to three scenarios. Consider a first scenario where the user visualizes a dataset with a high-resolution, large field of view conventional image. The user can select image regions of interest and modify the viewpoint for each individual region interactively to alleviate occlusions. The overall view does not change and it provides continuous context to connect the multiple focus regions (Fig. 2.1). This scenario is supported with a *top-down* multiperspective camera constructor that modifies the ray bundle of each focus region. Top-down construction supports multiple disjoint focus regions (Fig. 2.4).

Consider a second scenario where a user explores a dataset through conventional interactive visualization; views of interest are saved as they are encountered, and then they are integrated seamlessly into a multiperspective visualization (Fig. 2.2). The views of interest appear undistorted as subregions of the multiperspective image.

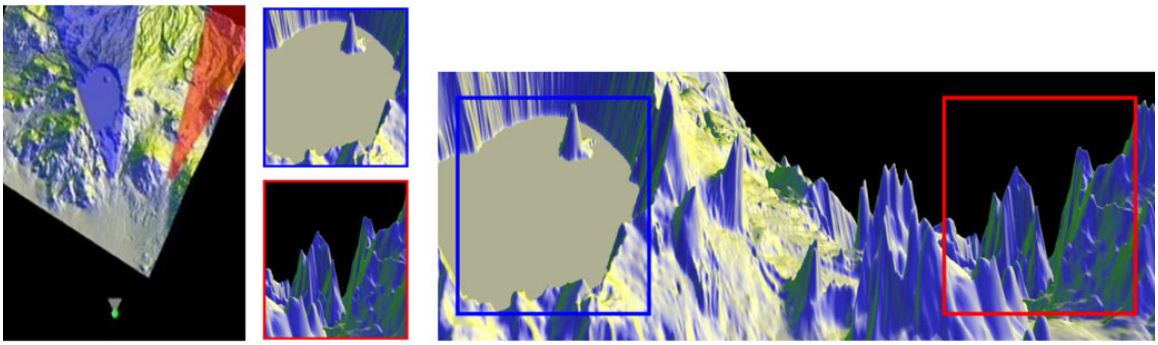


Fig. 2.2.: Multiperspective visualization that integrates two input views. The user explores the dataset interactively and selects two views of interest (overhead view, left, and views of interest, middle); then the system connects the two views seamlessly in a multiperspective image.

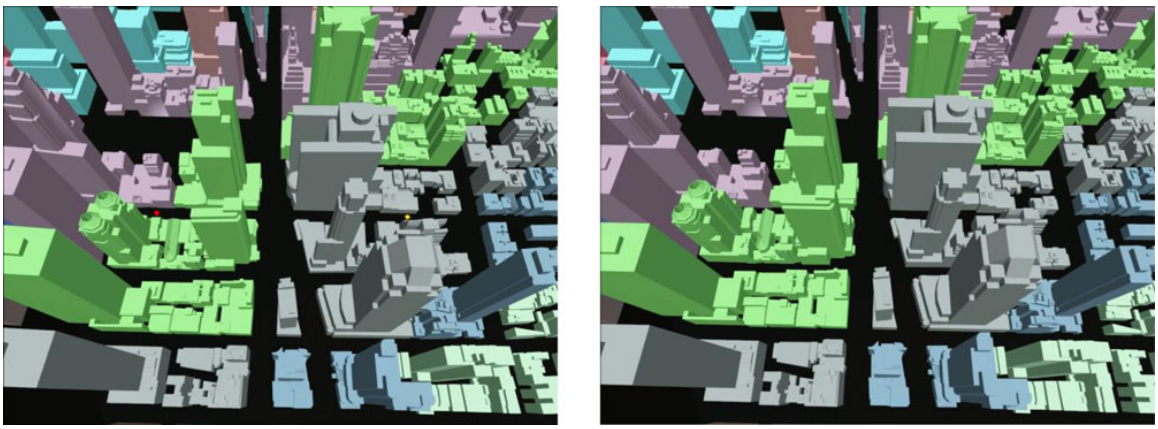


Fig. 2.3.: Multiperspective visualization (left) of an urban dataset showing two targets (red and yellow dots) that are occluded in a conventional visualization (right). The targets are shown where they would be visible in the conventional visualization in the absence of occluders.

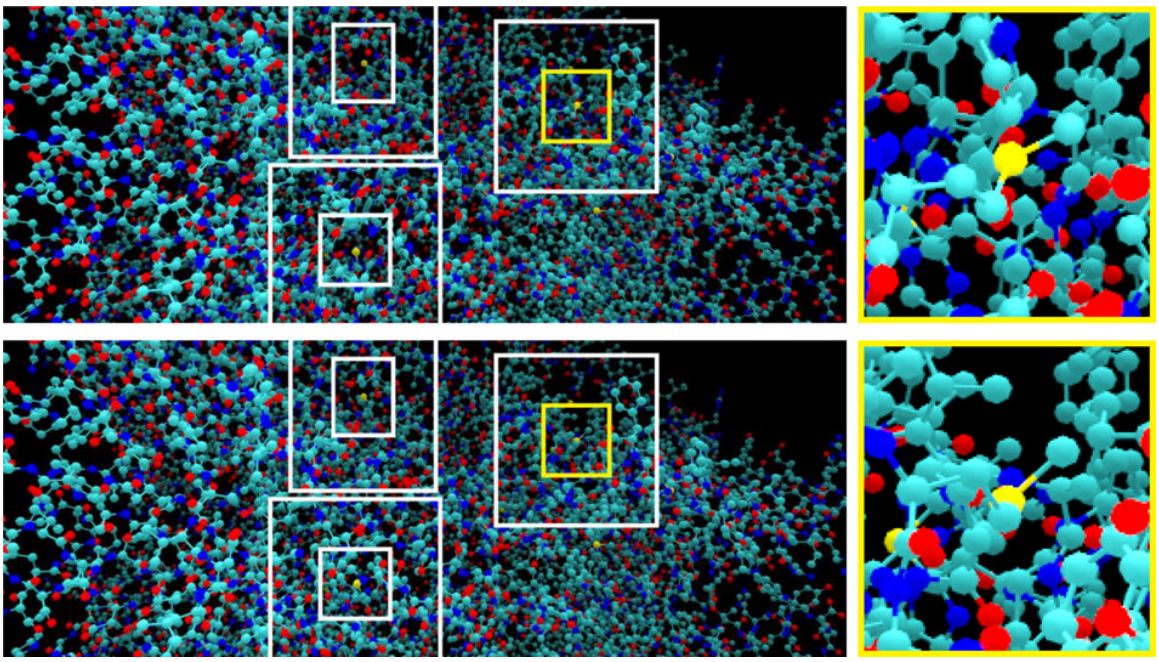


Fig. 2.4.: Multiperspective visualization of a protein molecule dataset with three focus regions (top), and conventional visualization (bottom), for comparison. The focus regions have different viewpoints than the overall visualization, revealing the two bonds of the yellow atom.

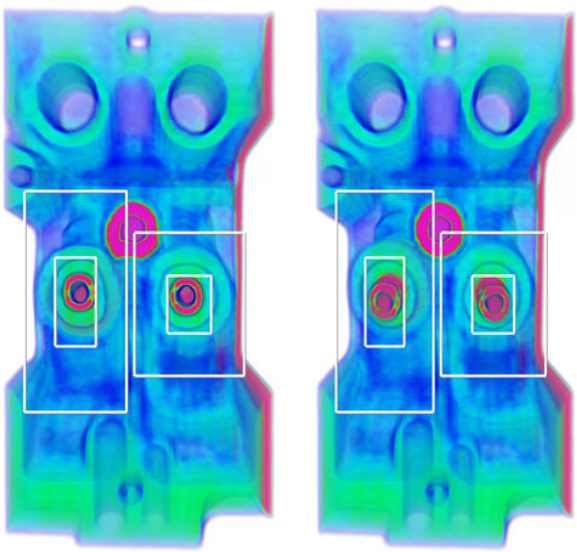


Fig. 2.5.: Multiperspective volume rendering of a density engine block dataset (left) and conventional visualization (right), for comparison. In the multiperspective visualization the viewpoints of the two focus regions are adjusted to provide a visualization along the piston axes.

This scenario is supported with a *bottom-up* multiperspective camera constructor that integrates conventional views.

Consider a third scenario where the goal is to avoid occlusions to one or several data subsets of interest, i.e., targets. As the targets move, the visualization adapts automatically to keep the targets visible (Fig. 2.3). The targets are shown where they would be visible in the conventional image in the absence of occluders, which conveys to the user the correct direction to the target. When the targets move to locations where they become visible, the multiperspective visualization reverts automatically to a conventional visualization, which can be used as is since it does not suffer from occlusion.

We demonstrate our multiperspective framework in the context of terrain (Figs. 1 and 2), urban (Fig. 2.3), and molecular biology (Fig. 2.4) datasets. Since our multiperspective visualization approach is based on an intervention on the underlying camera model, our approach ports straightforwardly to volume rendering of density datasets by ray casting (Fig. 2.5).

2.2 Prior Work

The more 3D datasets grow in complexity, the more occlusions stemming from the single viewpoint restriction of conventional images become a limiting factor in visualization. Occlusions are an open research problem that has been approached from many directions [6]. We limit the discussion of prior work to deformation and multiperspective techniques that are most relevant to our approach.

Gaining an unobstructed line of sight to a data subset of interest can be done by deforming the dataset. The idea was first used in the context of 2D data visualization, e.g., in the context of graph visualization [7], and it was subsequently extended to 3D datasets, e.g., in the context of short route [8] or car navigation [9], [10] visualization. The goal is to achieve the desired occlusion alleviation by deforming the dataset as little as possible. The deformation approach is the dual of the multiperspective approach. Visualizing a distorted dataset with a conventional camera can be seen as visualizing the original dataset with a multiperspective camera. For example, the multiperspective image in Fig. 2.3 could be obtained by distorting the urban dataset and then visualizing it with a conventional camera. For volumetric datasets, we can alternatively distort the intermediate sampling agents [11]. The difference is that multiperspective visualizations are constructed by modifying the underlying camera model. The occlusion alleviation effect is designed with great control directly in the image domain, as opposed to indirectly in the 3D dataset.

Multiperspective visualization originates in art, where the single viewpoint constraint is occasionally abandoned in the interest of artistic expression, effects that were replicated by computer graphics systems [12]. One practical application is to generate panoramas that alleviate occlusions and emphasize important landmarks and terrain features as needed for example for hiking and skiing maps [13], [14], [15]. This prior work handles height-field terrain dataset, with regions of interest known a priori, and it automates the expertise of professional illustrators. Multiperspective images have also been used to integrate photographs taken from different viewpoints (e.g., street

panoramas [16], [17]). In 2D animation, a single multiperspective panorama provides an animation sequence by sliding the frame rectangle over the panorama on a predefined path [18]. Both street and 2D animation panoramas are limited to special scenes.

Compared to these panorama techniques, our approach allows for interactive exploration of the dataset in search of regions of interest that might not be known a priori. Furthermore, the datasets are not restricted to height fields, and the focus regions are imaged without distortion. Like conventional focus+context visualization, we have the goal of an unequitable distribution of image samples in favor of the focus regions, whereas panoramas excel at pursuing visualization comprehensiveness.

The generality and flexibility of multiperspective visualization increased through innovations at camera model level. Multiple center of projection images [19] are obtained by rendering a 3D dataset with a one-column camera that slides along a path (i.e., a push-broom camera). Samples from thousands of viewpoints are integrated into a continuous image. The user has good control over viewpoint selection by designing the acquisition path. However, rendering the image is too expensive for interactive visualization or time-varying datasets, as it involves one rendering pass for each image column. Our multiperspective rendering framework relies on a more efficient parameterization of the ray space based on a few (i.e., 10-20) camera segments, which provides the needed occlusion alleviation flexibility without sacrificing rendering performance.

The general linear camera (GLC) [20] is at the other end of the multiperspective camera complexity spectrum by possibly being the simplest non-pinhole camera. The rays of the GLC are obtained by interpolating three input rays, and in the case of multiple GLC's, by interpolating the rays from all GLC's which overlap on the image plane [21]. The GLC provides fast projection, which ensures rendering efficiency. Given a 3D point inside the GLC frustum, one can compute the barycentric coordinates of the point's projection on the triangular GLC image by solving linear equations. However, the original parameterization of rays does not provide continu-

ity between adjacent GLC's that share two construction rays. A continuous GLC parameterization has been subsequently proposed [22], which comes at the cost of cubic projection equations. A single GLC doesn't have the occlusion alleviation capability needed in multiperspective visualization, but we use continuous GLC's in our framework to model camera segments, as described in Section 2.3.

Occlusion cameras are a family of non-pinhole cameras designed to extend the viewpoint of conventional cameras to a view region [23]. The camera rays are bent at occluder silhouettes to capture "barely hidden" samples, which are samples that become visible for small viewpoint translations. The resulting image is a high-quality aggressive solution to the from-region visibility problem, i.e., it finds most but not all visible samples. Occlusion cameras generalize the viewpoint to a continuum of nearby viewpoints, whereas what is needed for multiperspective visualization is a generalization of the viewpoint to a small set of distant viewpoints.

The needed viewpoint generalization is provided by the graph camera [24]. Starting from a conventional planar pinhole camera, the graph camera is constructed through a series of frustum bending, splitting, and merging operations applied recursively. The resulting camera is a graph of PPC segments. The piecewise linear rays are designed to circumvent occluders and to reach far into the dataset. Compared to the graph camera, the camera employed in our multiperspective framework is built from a mix of PPC and continuous general linear camera (CGLC) segments. The CGLC segments enable frustum splitting while maintaining image continuity, thereby overcoming a major shortcoming of the graph camera (see Section 2.4.1). Graph cameras can only be constructed automatically using a 2D maze with right angle intersections as scaffold, whereas in our framework the multiperspective camera is built automatically, in 3D, to track targets or to integrate conventional input images. Finally, the graph camera does not allow controlling where a data subset of interest is imaged, whereas our framework allows imaging a subset where it would be imaged in a conventional visualization. This enables the user not only to examine but also to locate the subset of interest.

Inspired by the nonlinear trajectory of light in proximity of large masses, multiperspective visualizations have been proposed based on curved rays [25], [26], [27], [28]. Curved rays have later been used for visualization outside of astronomy using ray segments connected with Bezier arcs [29]. The advantage of curved rays is that the transition from one viewpoint to the next is gradual, which reduces the distortion for objects that are imaged from more than one viewpoint. The cost is a lower rendering performance due to the higher camera complexity. Localized curved ray traversal [30] alleviates this problem, but is unsuitable for deforming the context region. The curved ray camera is a 1D sequence of camera segments and it does not support splitting, which limits its applicability to removing the occlusion for a single target. We use piecewise linear rays, with C_0 continuity, but the C_1 continuous rays of the curved ray camera [29] could be integrated into our framework for applications where the additional cost is warranted.

In conclusion, prior work has demonstrated the potential of flexible visualizations that enable a local changes to resolution and viewpoint. The work presented in this chapter is inscribed in the multiperspective visualization line of research. Our contributions include (1) a novel multiperspective camera model that generalizes the graph camera with CGLC nodes, which enables sampling the entire viewing volume continuously and non-redundantly, while still providing the fast projection operation essential to efficient feed-forward rendering; (2) an interactive camera constructor that allows the user to define several nonoverlapping focus regions and to tune the viewpoint for each focus region independently, while maintaining visualization continuity to the context region; (3) a top down and a bottom up camera constructor that allow splitting a root PPC into multiple perspectives as well as merging several PPCs into one common perspective; (4) a target tracking constructor that builds the camera model to ensure data subsets of interest remain occlusion free as the visualization camera or the data subsets move.

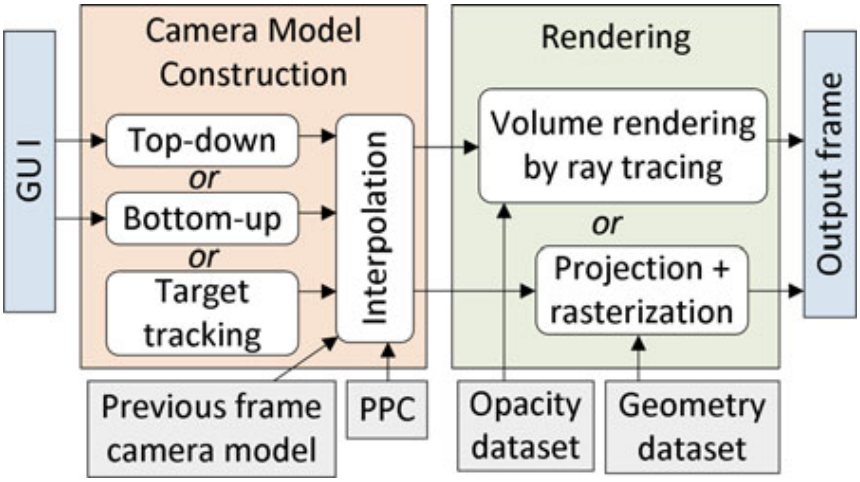


Fig. 2.6.: Overview of our multiperspective interactive visualization framework. For each frame, the multiperspective camera model is constructed first, in one of three ways, and then the multiperspective image is rendered.

2.3 Multiperspective Camera

Fig. 2.6 gives an overview of our framework. For each frame, the multiperspective camera model is constructed first. The camera model can be constructed with user input using a graphical user interface. The camera model can also be constructed automatically in top-down fashion, in bottom-up fashion, or to track data subsets of interest, i.e., targets. The camera model is interpolated with the camera of the previous frame to limit the magnitude of frame to frame camera model changes or with a conventional PPC in order to revert the multiperspective visualization back to a conventional visualization. Then geometry datasets are rendered by multiperspective projection followed by conventional rasterization, and density datasets are volume rendered by ray casting. In this section, we first describe the multiperspective camera model, we then describe how 3D datasets are rendered with the camera to obtain multiperspective images, and finally we describe how the camera is constructed in one of three ways to support interactive multiperspective visualization.

2.3.1 Camera Model

A camera model is a function that assigns a ray to each image plane sampling location. We have designed a multiperspective camera model based on the following considerations.

Flexibility; the camera should be able to integrate conventional PPC frusta in order to generate a multiperspective image that shows subsets of interest undistorted, each from its own viewpoint. Moreover, the camera should support a smooth, gradual change of the perspectives it integrates to enable effective interactive exploration of a dataset where users change the regions on which they focus, and to enable the visualization of a dynamic dataset, where the regions of focus can move over time.

Continuity and non-redundancy; the camera rays should sample the entire space subtended by the regions of interest, without gaps, in order to connect the images of the regions of interest with continuous context. Like in conventional focus+context visualization, continuity is important to show correctly the connection of the focus region to the context. Visualization discontinuity would place a significant additional cognitive load on the user who has to establish mentally the connection between the focus region and the context. Each point in the sampled space should be sampled by exactly one ray. This implies rays cannot intersect or, equivalently, the camera model projects one point in space to one point on the image.

Projection efficiency; given a 3D point inside its frustum, the camera model should provide a fast method for computing the image plane projection of the point in order to support efficient rendering on the GPU by projection followed by rasterization.

Fig. 2.7 gives a 2D illustration of our multiperspective camera model. A root PPC segment a, with viewpoint V_0 is used to integrate two leaf PPC segments, b and c with viewpoints V_1 and V_2. Segments b and c sample the data subsets of interest; they are connected to a with camera segments e and f, each implemented with two CGLCs.

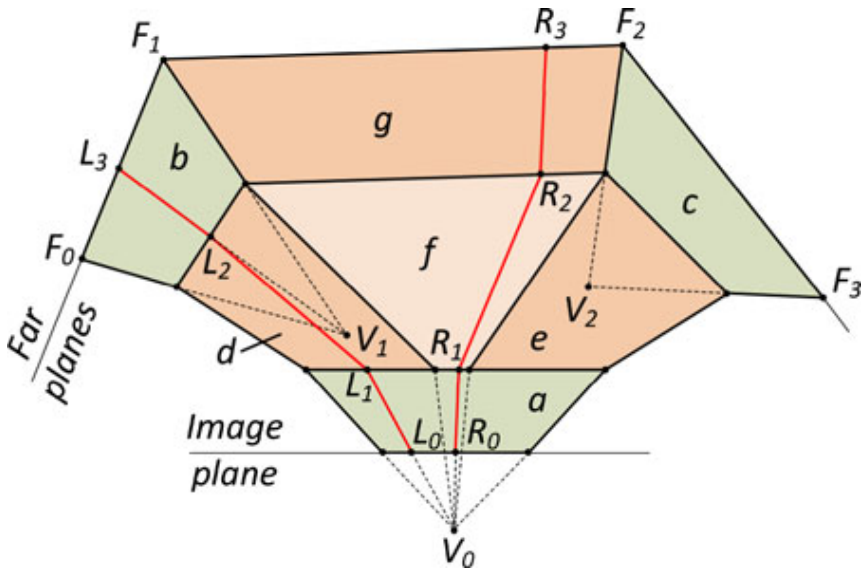


Fig. 2.7.: Camera model that integrates two conventional images with viewpoints V_1 and V_2 into a multiperspective image. The green camera segments are implemented with conventional planar pinhole cameras. The orange camera segments are implemented with continuous general linear cameras. The rays are piecewise linear, e.g. $L_0L_1L_2L_3$ and $R_0R_1R_2R_3$.

Using Fig. 2.7 again, camera segments f and g sample the dataset in between the subsets of interest to connect the two images with continuous context. Each of them is implemented with two CGLCs. The camera has piecewise linear rays. Ray $L_0L_1L_2L_3$ has three segments, one for each of the camera segments it traverses. Line L_2L_3 passes through V_1, and line L_0L_1 passes through V_0. The far planes F_0F_1, F_1F_2, and F_2F_3 define the far boundary of the camera. CGLC camera segments are the simplest camera segments that tile seamlessly both laterally and longitudinally. A CGLC segment can act as a wedge in between two conventional camera segments to achieve a smooth transition. CGLC segments can be stacked to change ray direction as needed for occlusion alleviation.

The camera model allows tuning the percentage of the multiperspective image pixels allotted to each subset of interest (b and c) and to the context (g), by changing how many of the rays of a are routed to each of the connecting camera segments d, f, and e. Fig. 2.7 shows a typical case where the context g is sampled at low resolution in favor of the subsets of interest b and c. The multiperspective camera can morph to a conventional PPC by straightening its piecewise linear rays to become single line segments. This is achieved by gradually translating V_1 and V_2 to V_0 and by aligning the connecting camera segments d, f, and e with rays from V_0 (Fig. 2.7).

In summary, our multiperspective camera is a graph camera [24] where some of the camera segments are CGLC [22] segments. The CGLC segments allow sampling the entire viewing volume continuously and non-redundantly.

2.3.2 Rendering by Projection & Rasterization

A 3D dataset modeled with triangles is rendered on the GPU by projection followed by rasterization.

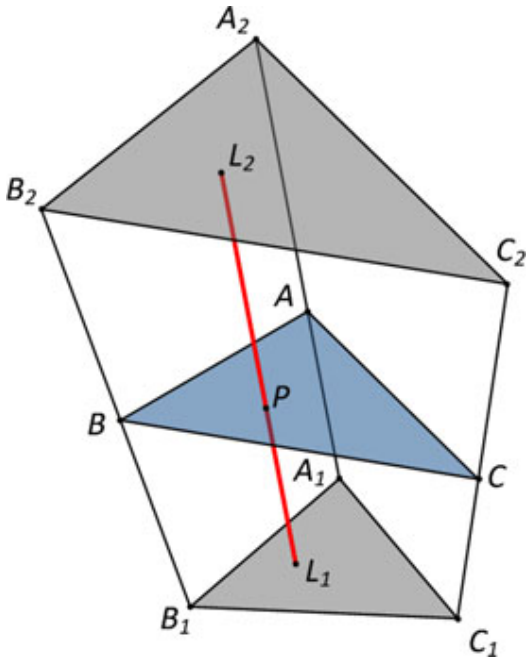


Fig. 2.8.: Continuous general linear camera frustum. The endpoints L_1 and L_2 of ray L_1L_2 have the same barycentric coordinates in triangles $A_1B_1C_1$ and $A_2B_2C_2$. 3D point P projects at L_1.

Projection

Given a 3D point P, the point is projected with each camera segment until a valid projection is found. A projection is valid if P is inside the frustum of the camera segment. PPC segments use the conventional projection. Fig. 2.8 illustrates the projection of a point P with a CGLC frustum. The CGLC rays, which have non-concurrent lines, travel from $A_2B_2C_2$ to $A_1B_1C_1$. Projection has to find the ray through P, to compute the intersection L_1 of the ray with $A_1B_1C_1$, and then to map L_1 to the multiperspective output image. First we find a plane through P that splits the CGLC construction ray segments A_1A_2, B_1B_2, and C_1C_2 in the same ratio t:

$$t = \frac{A_1A}{A_1A_2} = \frac{B_1B}{B_1B_2} = \frac{C_1C}{C_1C_2} \tag{2.1}$$

Parameter t is computed by solving a cubic equation. Once t is known, points A, B, and C are known, and one can compute the barycentric coordinates (α, β, γ) of P in triangle ABC. (α, β, γ) are found by inverse barycentric interpolation, which implies solving a quadratic. Then the projection L_1 of P onto $A_1B_1C_1$ is computed as:

$$L_1 = \alpha A_1 + \beta B_1 + \gamma C_1 \tag{2.2}$$

The projection is valid iff $0 \leq t \leq 1$ and the barycentric coordinates (α, β, γ) satisfy $\alpha, \beta, \gamma \geq 0$ and $\alpha + \beta + \gamma = 1$. The continuity condition requires that neighboring CGLC frusta tile seamlessly, and the non-redundancy condition dictates that there can be at most one valid projection for each point in the data space. This implies that no two frusta intersect, and all frusta are convex.

Once a valid projection is found, the projected point L_1 can be projected recursively with each camera segment upstream along the path to the root segment. The final barycentric coordinates then linearly map to the multiperspective output image coordinates for P. However, one does not need to project recursively the point P. Instead, we map the vertices of the near face of each camera segment to the multiperspective output image during camera construction. Using Fig. 2.8 again, let (u_A, v_A), (u_B, v_B), and (u_C, v_C) be the output image coordinates of A_1, B_1, and C_1. Then the output image projection (u, v) of P is computed as:

$$(u, v) = (\alpha u_A + \beta u_B + \gamma u_C, \alpha v_A + \beta v_B + \gamma v_C) \tag{2.3}$$

Rasterization

The projection of a triangle contained by a CGLC segment has curved edges. We approximate CGLC rasterization with conventional rasterization and we control the approximation error by subdividing any large triangle offline. Visibility is computed using the regular depth test. The z value used is the fractional parameter t that locates the 3D point within its camera segment, plus the camera segment index i.

Camera segments are depth indexed from the root to the leaf segments. In Fig. 2.7, a has depth index 0, d, f, and e have depth index 1, and b, g, and c have depth index 2.

2.3.3 Rendering by Ray Casting

For some visualization techniques, such as volume rendering [31], the multiperspective image cannot be computed by projection followed by rasterization. To support such techniques, one has to be able to compute the piecewise linear ray for a given image plane location. Computing the ray is similar to the projection operation, but this time we move downstream from the root to the leaf camera segments. Using Fig. 2.7 again, given image plane point L_0, the first segment L_0L_1 of the ray is computed by intersecting V_0L_0 with the far face of camera segment a. Then, L_2 is computed on the far face of d using the barycentric coordinates of L_1 on the near face of d. This defines the second ray segment L_1L_2. Finally, b is a PPC segment and the third ray segment L_2L_3 is computed by intersecting V_1L_2 with the far face F_0F_1 of b.

2.3.4 Camera Construction

We have developed three methods for constructing the multiperspective camera.

Bottom-Up Construction

In bottom-up construction, the user navigates a conventional PPC through the dataset using the keyboard for translation and rotation commands. As views of interest are found, the user saves them as the PPC segments b and c (Fig. 2.7). The relationship between the views of interest is conveyed with an overhead visualization (left image in Fig. 2.2, also see video). Once both b and c are saved, the system constructs the remaining segments automatically as follows. V_0 is positioned first in front of the near faces of b and c. The field of view of a is chosen to encompass the

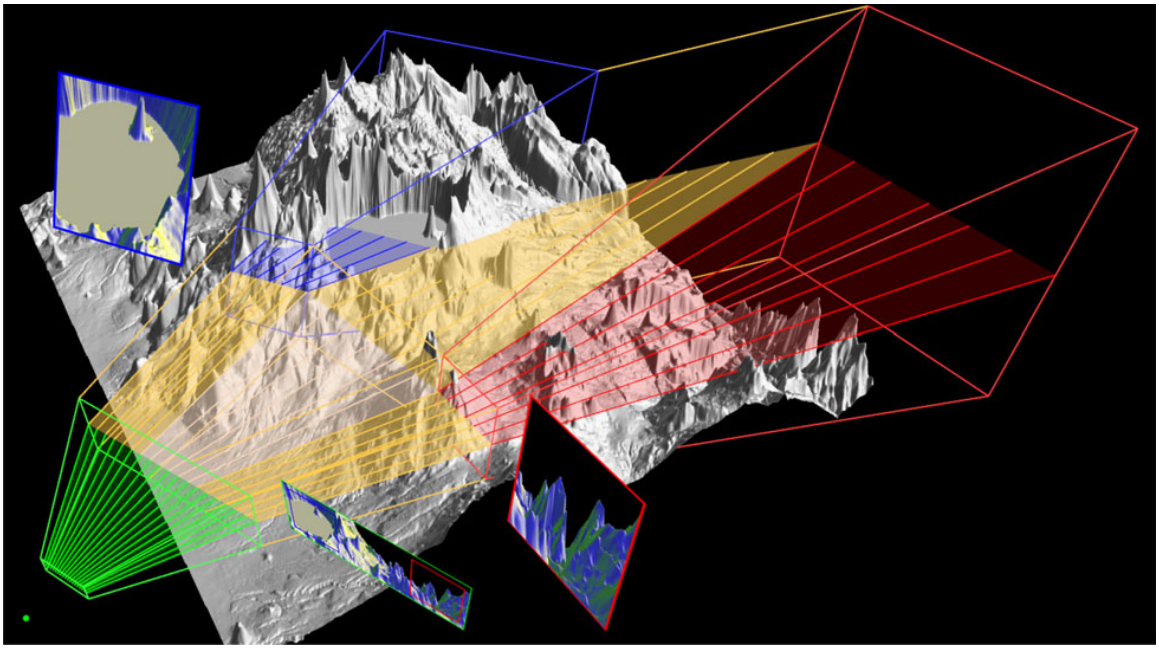


Fig. 2.9.: Multiperspective camera model with the structure shown in Fig. 2.7 constructed in bottom-up fashion for the scenario shown in Fig. 2.2. The green, blue, and red frusta correspond to camera segments a, b, and c in Fig. 2.7.

near faces of b and c and to capture additional foreground and background according to user specified parameters. Finally, segments f and g are constructed to bridge the gap between (d, e) and (b, c). Fig. 2.9 illustrates the multiperspective camera model from Fig. 2.7 specialized for the scenario shown in Fig. 2.2. Only the central row of rays is shown.

Top-Down Construction

The user starts by navigating a conventional pinhole camera that provides an overview of the dataset. The user can explore a data subset in more detail as follows. First, the user clicks on a dataset point C that will serve as the center of the focus region (Fig. 2.10). Then, the user constructs a rectangular focus region S centered at C. The focus region defines a camera segment that corresponds to a bundle of rays of the overview camera. The user rotates the camera segment interactively to change

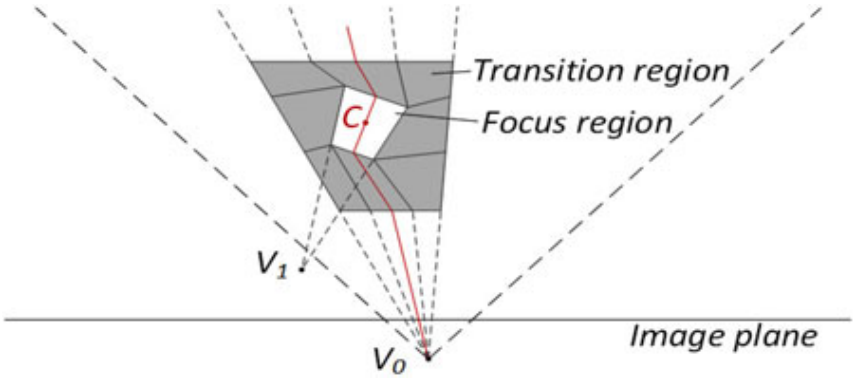


Fig. 2.10.: Top-down construction around point C, the center of focus region S. The lines of the bundle of rays in the focus region converge at viewpoint V_1, which rotates around C by user input. The transition region (grey) gradually reverts the perspective to the main viewpoint V_0.

perspective on the focus region. The revolution of the camera keeps the focal point C in the center of the focus region. The focus region is padded with a transition region over which the perspective reverts gradually and continuously from the perspective of the camera segment to the main perspective of the overview camera (Fig. 2.4).

Target Tracking Construction

The construction of the multiperspective camera for tracking a single target is illustrated in Fig. 2.11. The target T is occluded from V_0. In the absence of the occluders, T would be visible at P. The construction algorithm reroutes the rays of the pixels around P to go around the occluders. This way the target is visible in the multiperspective image at P, and the occluders are "pushed downwards" (i.e., to the bottom of the image). The rays are rerouted using four camera segments a, b, c, and d. The neighboring camera segments (grey) transition back to the planar pinhole camera, encapsulating the perturbation needed for target tracking. All camera segments have parallel near and far base planes. For projection, plane ABC (Fig. 2.8) is simply constructed parallel to the base planes, which saves having to solve the cubic equation

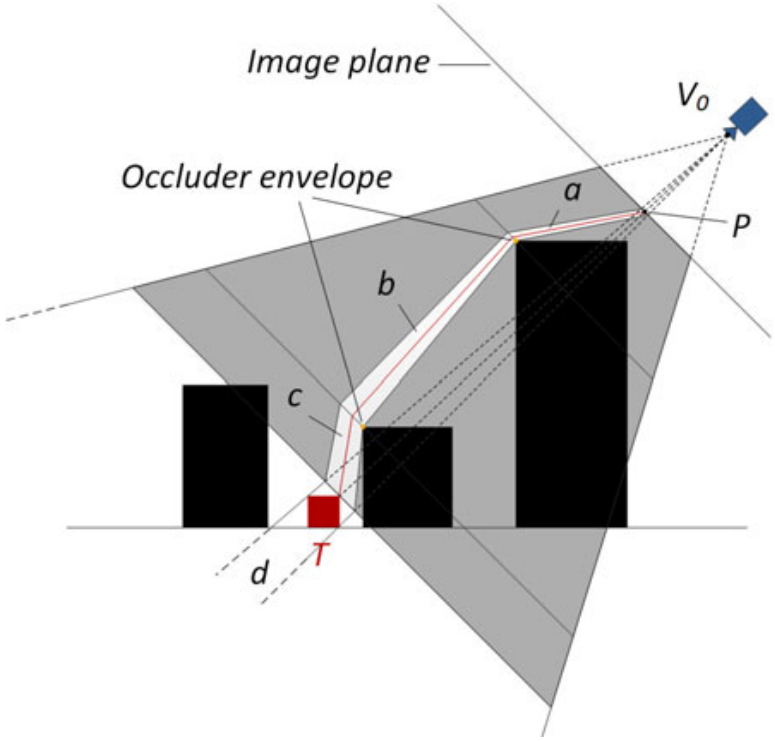


Fig. 2.11.: Multiperspective camera avoiding the occlusion of target T. T would be visible at P in the absence of occluders. The bundle of rays around P is routed just clear of the two occluders (tall and short black rectangles) hiding T.

to find parameter t (see the Projection Section 2.3.2). Segment d has the original viewpoint V_0 so T is seen as it would be seen in the absence of occluders.

Multiple targets can be tracked at once, and the targets can converge and then diverge again (see Fig. 2.3 and video). We *do not* construct one complex multiperspective camera that removes occlusions to all targets. Rather we construct one multiperspective camera for each target independently and compute the final projection of a vertex by projecting the vertex with each camera and by computing a weighted average of the preliminary projections. The weight of a preliminary projection is based on how much the projection is shifted from where the vertex would project with a conventional PPC. Larger shifts correspond to larger weights. When targets do not overlap in the image plane, the projection of a vertex is affected by at most one target, and the vertex is projected as in the case of a single target. When

two targets overlap, the preliminary and final projections of a vertex at the region of overlap are the same.

2.4 Results and Discussion

We have tested our approach for managing occlusions in three scenarios and with several datasets. We first discuss the occlusion alleviation capability of our approach. Then we discuss the multiperspective rendering performance achieved, which is important for interactive visualization. Finally we discuss limitations.

2.4.1 Quality

The major design concerns for our multiperspective visualization framework are visualization construction flexibility and image continuity. As shown in the images in this chapter, flexibility has been achieved through a camera model that allows modifying the viewpoint for individual regions of a given image, connecting conventional images with continuous context, and tracking one or more targets while avoiding occlusions.

Fig. 2.12 shows that our framework can connect two conventional images with opposite views into a continuous image, whereas a similar image rendered with the prior work graph camera framework [24] suffers from discontinuities which cannot always be hidden behind geometry. As shown in the accompanying video, in a graph camera visualization, an airplane flying above the city disappears and reappears twice as it crosses the discontinuities. In our visualization, the airplane is visible at all times. We achieved this improvement by using CGLC frusta that can perform a split of perspective while sampling the entire space in between the outgoing branches without leaving any void; the PPC frusta used by the graph camera cannot perform the split operation without voids, which result in visualization discontinuity.

Fig. 2.13 shows that our framework can reveal a target hidden in a heavily occluded dataset with only minimal image distortions. The framework supports mul-

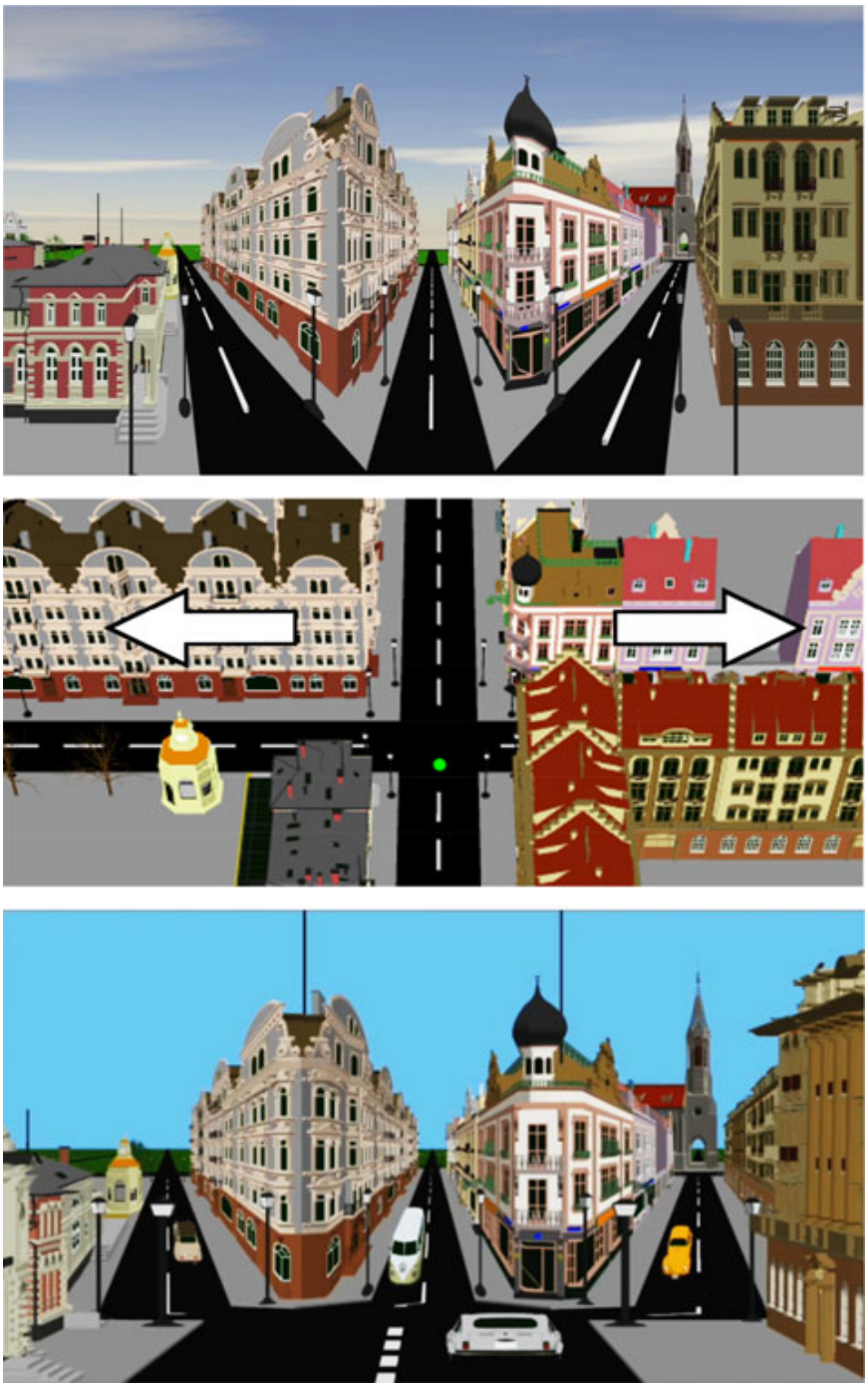


Fig. 2.12.: Multiperspective image rendered with our framework (top) that integrates two opposite views (middle) and image rendered with the prior art graph camera framework for comparison (bottom). Our image visualizes the entire sky, whereas the graph camera image misses parts of the sky at the two vertical discontinuities shown with black line segments.

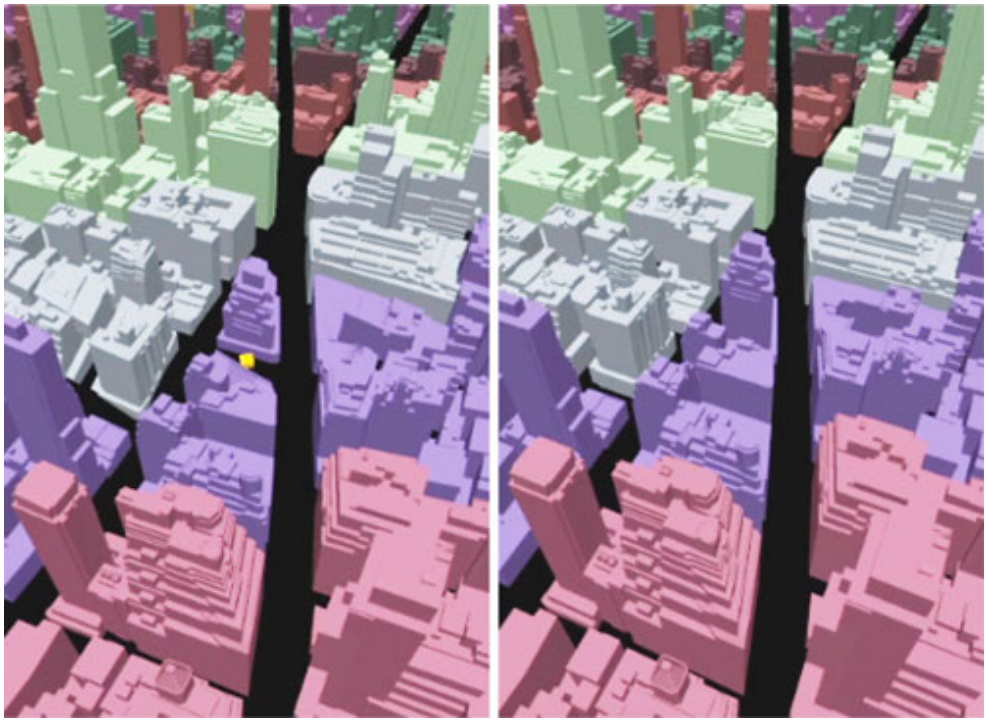


Fig. 2.13.: Multiperspective image rendered with our framework that tracks a target in a heavily occluded dataset (left); conventional image with hidden target, for comparison (right).

tiple moving targets which can converge to the same image region and then diverge again. Compared to the curved ray camera [29], which supports target tracking only through sequential multiple viewpoints, our target tracking framework brings two improvements. First, by using flexible CGLC frusta which tile laterally, one can connect the target tunnel to the undistorted peripheral frustum seamlessly, as shown by the grey shaded area in Fig. 2.11. Second, the flexible CGLC frusta allow projecting the target itself and its background onto the image plane at the exact undistorted position and scale, as shown in Fig. 2.3. The curved ray camera visualizes the target at a completely different image location than where the target would be visible in the absence of occluders.

Our method is part of the general multiperspective class of occlusion managing techniques. As described in the prior work section, a dual approach for managing occlusions is to deform the dataset such that when viewed with a conventional camera

the regions of interest are not occluded. Once the multiperspective camera is defined, one could compute a distorted dataset as follows. Consider dataset vertex V that projects with the multiperspective camera at pixel p. Let z be the depth of V along the multiperspective ray that projects it. Compute the deformed position V' of V by unprojecting p at z with a conventional camera (i.e., along a straight ray). Rendering the deformed dataset with a conventional camera produces the multiperspective image.

One advantage of the multiperspective approach over the deformation approach is in a scenario where the targets for which occlusion should be avoided are not known a priori. In such a scenario, the multiperspective camera provides a convenient way of exploring parts of the dataset without requiring the user to navigate the main camera to each part. Our camera has the ability to extend "branches" to preview dataset regions without the user interface manipulation and the disorientation inherent to navigating a conventional camera around occluders and retracing one's steps. The piecewise linear rays simulate translating the camera to reveal the target, providing an intuitive preview of what would be seen if the camera were actually translated. A second advantage is that the leaf camera is a conventional camera so the target of the exploration is always visualized distortion free.

Out of all our examples, the target tracking in the urban dataset is probably the easiest to replicate with the deformation approach. Here the targets are known, and deformation would simply shrink the height of all buildings along the line of sight to the target. However, the buildings close to the target would have to be shrunk completely, to street level, which affects their recognisability. For the case in Fig. 2.13, the deformation approach would have to visualize the occluding building below the target with essentially only a contour of its roof, whereas our visualization achieves occlusion alleviation while conveying the height of the building (Fig. 2.13, left).

Our top-down constructor supports changing the viewpoint for any number of disjoint focus regions (Fig. 2.4), and the target tracking supports multiple dynamic targets that can overlap (Fig. 2.3, video). Our approach relies on modifying the rays

of the underlying camera used to compute the visualization. Therefore the framework can be leveraged in the context of most visualization techniques. For example, in the case of volume rendering (Fig. 2.5), the rays of the multiperspective camera can be traced through the volume dataset to integrate opacity like in conventional volume rendering.

Like any deformation or multiperspective visualization approach, our technique introduces distortions to make room for the samples of the region of interest that is occluded in a conventional visualization. The distortion is confined to the transition region. Even so, some applications might prefer that the parts of the visualization that deviate from a conventional planar pinhole camera visualization be clearly indicated to the user. We achieve this with one of two modes: a red highlight commensurate with the amount of distortion (Fig. 2.14, top) and an undistorted wireframe rendering of the parts of the dataset affected by the distortion, overlaid onto the multiperspective visualization (Fig. 2.14, bottom).

2.4.2 Performance

The timing information reported in this chapter was collected on an Intel Xeon E5-1660 3.3 GHz workstation with 16 GB of memory and with an NVIDIA Quadro K5000 4 GB graphics card. The implementation uses OpenGL and Cg [32] GPU shaders.

For datasets modeled with triangles, multiperspective rendering is implemented with a vertex shader that implements the multiperspective projection. The vertex is projected with each camera segment until a valid projection is found. Vertices outside the 3D axis aligned bounding box of the camera segment are trivially rejected. We start with the leaf segments (i.e., b, c, and g in Fig. 2.7), which are the largest and are therefore most likely to contain the vertex. For volume rendering, the fragment shader computes and traces the ray at the current pixel with even steps, integrating opacity and translating it to color using a transfer function.

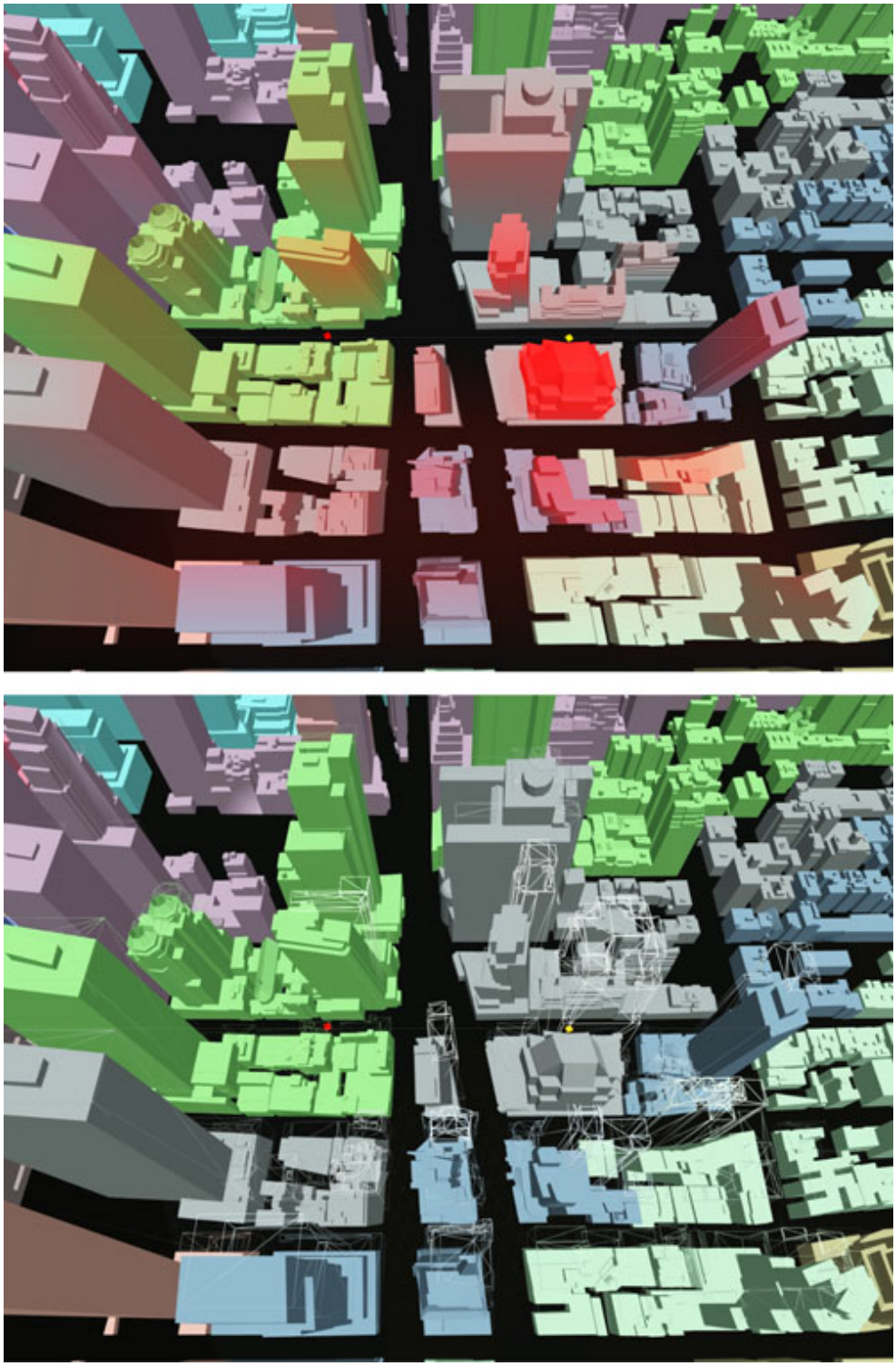


Fig. 2.14.: The amount of dataset distortion visualized by a red highlight (top) and by an undistorted wireframe rendering (bottom).

Table 2.1.: Multiperspective Rendering Performance

Dataset	Triangles	Resolution	View	Frame rate [fps]
Terrain	2,119,522	1,920×720	Fig. 2.1	46
			Fig. 2.2	30
Manhattan	3,691,768	1,280×960	Fig. 2.3	26
			Fig. 2.13	30
Eurotown	3,751,621	1,440×1,440	Fig. 2.12	28
Protein	1,438,992	1,920×720	Fig. 2.4	251
Volume	256×256×110	640×640	Fig. 2.5	8.9

The multiperspective rendering performance of our framework is given in Table 2.1.

We achieve interactive performance in all cases. Geometry rendering (all rows except for the last one) is much faster than volume rendering by ray casting (last row). However, volume rendering with a conventional PPC yields only a slightly better frame rate of 9.4 Hz, This confirms that the overhead of tracing along piecewise linear instead of linear rays is marginal. Rendering the Protein dataset is much faster than the other datasets because only a small fraction of dataset vertices project inside the focus regions, therefore the majority of the vertices are projected conventionally at a smaller cost. When the cubic projection equation has to be solved, we have found that solving the equation numerically is faster than evaluating the closed-form solution expression.

The nonlinear projection of the multiperspective camera implies that rasterization is also not linear. In other words, the nonlinear projection applies not only to the vertices of a triangle, but also to its interior. There are two major approaches to nonlinear rasterization, and we have experimented with both. One approach is to actually perform nonlinear rasterization in the fragment shader. For this, one first has to derive a method for approximating the image plane axis aligned bounding box of the projection of the triangle. Since the projected triangle now has curved edges, the approximation has to consider more than the vertices of the projected triangles. Then, nonlinear rasterization can be performed in 3D, by intersecting the dataset

triangle with the ray of each pixel of the axis aligned bounding box and interpolating rasterization parameters using the barycentric coordinates of the intersection point. A second approach is to approximate nonlinear rasterization with conventional rasterization but making sure the projected triangles are small enough. Online subdivision requires a geometry shader that issues a varying and potentially large number of primitives, which is a severe performance bottleneck. Offline subdivision has to be done in view independent way which can result in subdivisions that are either too coarse or to detailed for a particular viewpoint.

We have chosen the approach of offline subdivision. The cost of true nonlinear rasterization is unwarranted for the following three reasons. First, today's complex datasets are modeled with small triangles who can be rasterized conventionally with good results, without subdivision. Second, the focus regions are imaged with PPC segments where conventional rasterization is accurate, and nonlinear rasterization is only used for regions whose role is limited to providing context. Third, using conventional rasterization implies that there are no changes at the fragment shader, which makes our framework portable to any existing visualization effect.

Fig. 2.15 shows that the time needed to render a frame is linear with the number of focus regions. The time to render the dataset conventionally is $\sim$1.3 ms, and each focus region adds 0.8 ms.

2.4.3 Limitations

There is no fundamental limitation on the number of perspectives integrated using our framework. The top-down constructor supports any number of focus regions, and the target tracking constructor can handle multiple dynamic, possibly overlapping targets. However, all constructors presented here require that the camera segments be disjoint, which is harder and harder to achieve when the number of perspective increases. Overlapping camera segments translate to repeated visualization of the primitives located at the region of overlap, and such redundant visualization might

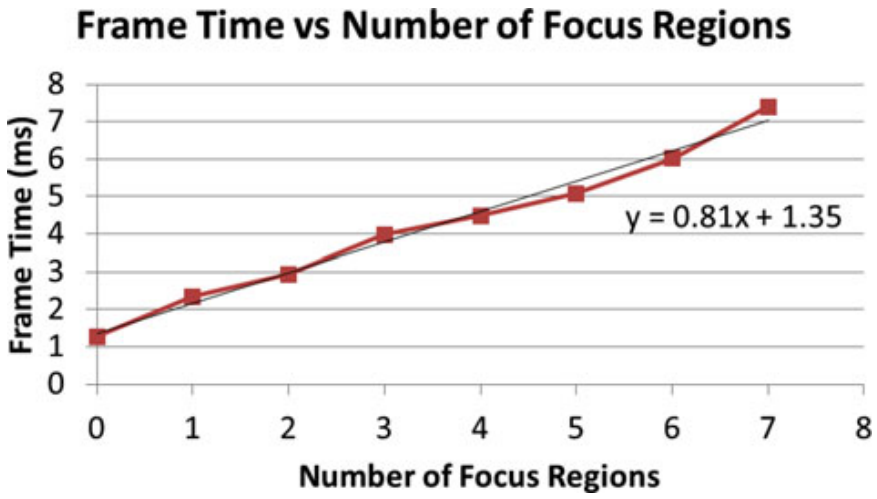


Fig. 2.15.: Frame time as a function of number of focus regions, for the Protein dataset shown in Fig. 2.4.

not be useful. Supporting redundant visualization with good performance is difficult because each vertex can have a variable number of projections. Moreover, each of three vertices of a triangle can have a different number of projections (i.e., some vertices are inside more camera segments than others), and projecting such a triangle requires clipping with individual camera segments.

The multiperspective visualization effectiveness decreases as the number of perspectives increases. First, the user interaction required to define camera models with many perspectives becomes complex. Second, the number of display pixels allotted to each perspective decreases, which limits the ability to convey clearly each perspective to the user. Multiperspective visualization trades image resolution for resolution along the view direction. Sufficient image resolution is needed for this trade-off to be beneficial. Third, the comprehensibility of the image decreases with the number of perspectives.

Another consideration that limits the number of perspectives is the decrease in performance brought by the increase in camera model complexity. Whereas for our examples good performance was obtained by projecting every vertex with every camera segment, scalability with the number of camera segments requires a faster method for determining the camera segment that contains a vertex. The frusta of CGLC segments do not have planar side faces therefore a scheme that subdivides space hierarchically using planes (e.g., a binary space partitioning tree or a kd-tree) will not separate two adjacent camera segments cleanly. Instead, one should use bounding volume hierarchies like the ones developed for ray casting acceleration. The goal is to achieve an $O(logs)$ projection time, where s is the number of camera segments.

Our multiperspective visualizations transition abruptly from one camera segment to the next. Even though the visualization is C_0 continuous, the abrupt transition can result in noticeable distortions (Fig. 2.16, video). We distinguish between two types of camera segment to camera segment transitions. A longitudinal transition is defined at the far face of an upstream camera segment that serves as the near face of the next, downstream camera segment, e.g., segment d to b in Fig. 2.7. At a longitudinal

transition, the piecewise linear rays break from the current linear segment to the next. A lateral transition is through a shared lateral face of two adjacent camera segments, e.g. d and f in Fig. 2.7. Rays do not traverse these lateral faces and are not affected by the lateral transition.

The distortion introduced by longitudinal transitions can be alleviated with two approaches. One approach, demonstrated by the curved ray camera framework [29], relies on rays modeled with Bézier arcs to transition from one viewpoint to the next, solution that can be adapted to our framework. A second approach is to subdivide longitudinally the connective camera segments (i.e., d, e, f, and g in Fig. 2.7), in order to achieve a gradual viewpoint change. The rays remain piecewise linear, but the ray segments are shorter which reduces the change in direction from one segment to the next.

The distortion introduced by lateral translations can be alleviated by subdividing a CGLC segment into a fan of CGLC segments. In Fig. 2.7, f would have to be subdivided into several segments, with the left-most sub-segment having a ray pattern very similar to camera segment d and with the right-most sub-segment having a ray pattern very similar to e.

Like any multiperspective or deformation approach, our visualization framework is not well suited for applications where it is important to convey global spatial relationships accurately. Our framework is well suited in an exploratory context where it saves the user to have to navigate around occluders in search of regions of interest or in contexts where the regions of interest are known and the main goal is to keep the targets visible, surrounded by context. Our framework shows the regions of interest free of distortions, as they are imaged with a conventional camera, and the distortions are confined to a transition region, beyond which the original visualization is not perturbed. Finally, an important design concern for the target tracking constructor is to image the target where it would be seen in the absence of occluders, correctly conveying the spatial relationship between the user and the target.

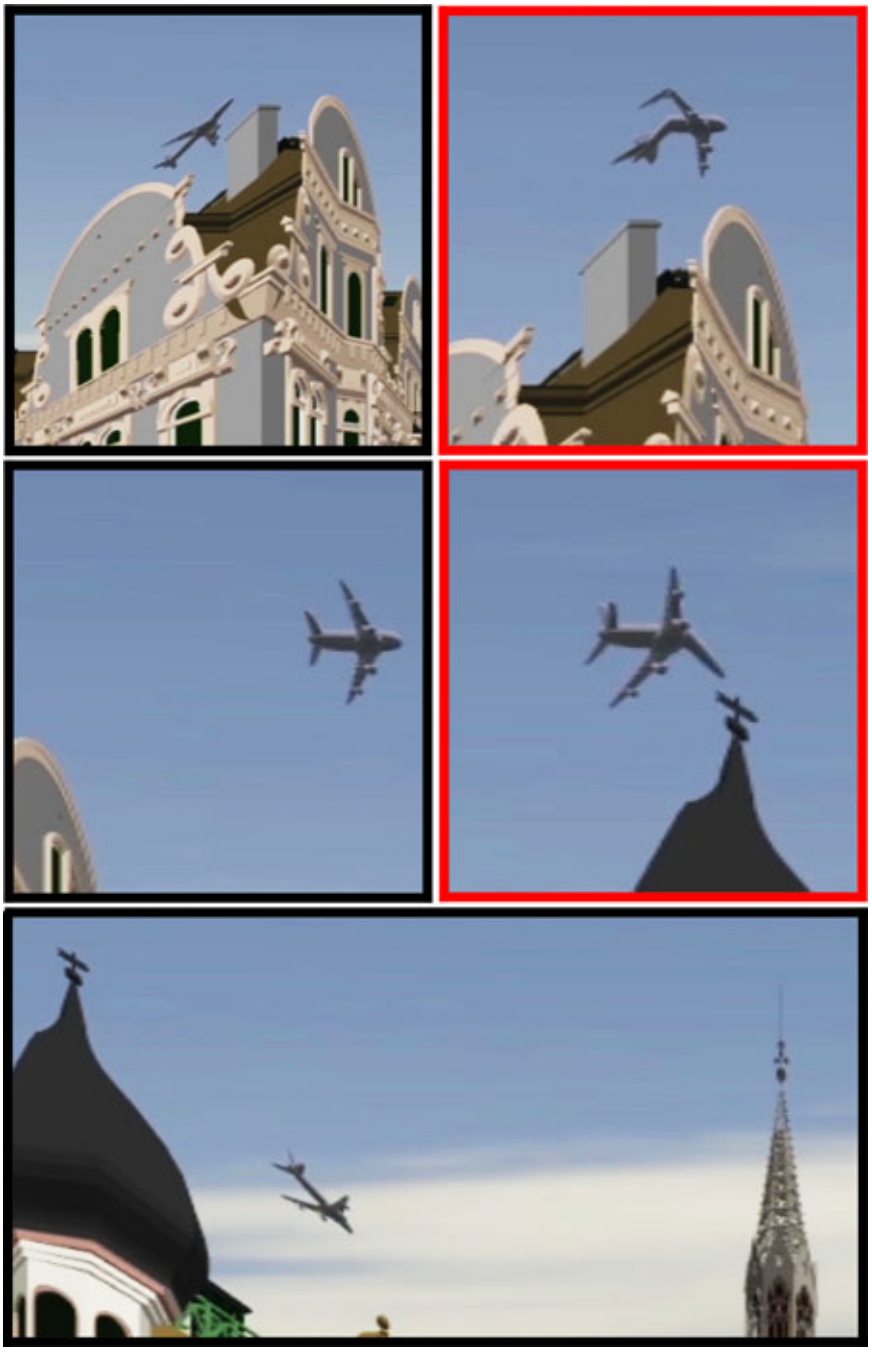


Fig. 2.16.: Fragments of the multiperspective visualization from Fig. 2.12 showing an airplane flying over the buildings. The airplane appears distorted as it crosses from one camera segment to the next (images with red border) and it is not distorted while completely contained by one segment (images with black border).

Another fundamental limitation inherited from the general class of multiperspective visualization approaches is the requirement that there is an unobstructed path from the user to the target, along which to route the rays to achieve occlusion alleviation. When such a path does not exist, e.g., in the case of an engine with internal parts, our approach has to be used in conjunction with other occlusion management techniques such as explosion [33], [34], cutaway [35], and transparency [36] approaches.

2.5 Conclusions and Future Work

We have presented a framework that advances the state of the art in multiperspective visualization: the framework allows constructing continuous multiperspective visualizations by changing the viewpoint for individual focus regions in an image, by connecting input images with continuous context, and by alleviating occlusions to moving targets without distorting or displacing the target subimages.

The framework relies on a flexible yet fast multiperspective camera. Whereas a conventional camera has a few parameters with which the application can interact directly (e.g., three rotations, three translations, focal length), our multiperspective camera comprises 10-20 camera segments which amounts to hundreds of parameters. The power of our framework comes from the three constructors that set all these parameters automatically to construct the desired multiperspective visualization. The constructors relieve the application from tedious low-level specification of the camera model, in favor of formulating high-level constraints that are satisfied automatically.

We have demonstrated our multiperspective visualization framework in the context of datasets modeled with triangles (i.e., terrain, urban, and bio molecular datasets) and in the context of volume rendered datasets. The framework is readily usable with datasets modeled with other types of geometric primitives, such as spherical particles, as long as the geometric primitives can be tessellated. Future work could focus on supporting higher order primitives without incurring the cost of full 3-D tessellation, for example by applying the multiperspective projection only to the particle

center and by resorting to a custom approximate rasterization. The examples shown in this chapter use rectangular focus regions. The smallest building block of our multiperspective camera is a CGLC which has a triangular frustum. Future work could examine defining focus and transition regions of more complex shapes based on polygons tessellated with triangles.

We see our work as making a visualization infrastructure contribution that enhances the information bandwidth of images. As such, we foresee that our work brings multiperspective visualization one step closer to becoming a tool for virtually all applications where visualization is needed. Our multiperspective visualization can increase efficiency in a visual search, where using a conventional visualization would require navigating the camera to a potential region of interest and then back, if the region turns out to be a false positive. For example, when examining a density dataset, a physician could change the viewpoint on a subvolume to better examine it, without losing context. A molecular biologist could examine the fit between a designed molecule and multiple receptor sites that cannot be all imaged with the same conventional visualization due to occlusions.

Another direction of future work is the extension to multiperspective visualization of real-world real-time datasets. Consider an urban scene captured with video cameras mounted at intersections, on cars, and on aircraft. The building geometry is known, for example from off-line LIDAR acquisition and conventional CAD modeling, like is the case for our *Terrain* and *Manhattan* datasets. The goal is to integrate the real-time video feeds into a multiperspective visualization that avoids occlusions for one or more regions of interest. The geometric model acts like a connection between the various video feeds, indicating the redundant parts of the feeds, as well as how to assemble the feeds in a continuous multiperspective video panorama.

Our work advocates abandoning the traditional rigidity of the images used in visualization in favor of flexible images that the user can optimize for each viewpoint, dataset, and application.

2.6 Acknowledgements

The authors thank Jian Cui and Paul Rosen for advising on their earlier work on multiperspective rendering. The protein molecule dataset [37] was obtained from RCSB PDB [38]. The sky cubemap texture was obtained from Roel Reijerse [39] and used under Creative Commons, Attribution-NonCommercial-ShareAlike 3.0 Unported license. http://creativecommons.org/licenses/by-nc-sa/3.0/. They acknowledge support by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A130016, and by the US National Science Foundation (NSF), through Grant 1217215. The opinions expressed are those of the authors and do not represent views of the Institute of Education Sciences, of the U.S. Department of Education, or of the NSF.

3. EFFICIENT VR AND AR NAVIGATION THROUGH MULTIPERSPECTIVE OCCLUSION MANAGEMENT

3.1 Introduction

Tracked HMDs provide an intuitive interface for exploring virtual and real 3D scenes in VR and AR applications. The user naturally selects the desired view by walking and by rotating their head. However, the efficiency of such 3D scene exploration is limited by occlusions. Consider the case of a VR exploration of a city model with the goal of finding a specific street-level ROI. Tall buildings occlude the streets and the user has to move considerable amounts to gain a direct line sight to the street currently being examined. If the street proves to be empty, the user proceeds with examining the next street. This sequential exploration is inefficient. In the case of a dynamic ROI, such a sequential exploration might never find the ROI.

Consider the application of surveillance of corridors inside a building. Occlusions prevent the user from seeing beyond the current corridor segment. The user has to walk to each intersection sequentially in order to examine side corridors. Again, the sequential exploration might never find a moving intruder. Even worse, if the intruder is aware of the user's position, the intruder can easily avoid being detected. Another challenge of conventional navigation in VR and AR applications is that some of the viewpoints best suited for alleviating occlusions are unreachable. Consider the VR urban model mapped to a room. Walls and furniture might prevent the user from assuming the viewpoint that establishes a direct line of sight to a ROI. For example, a higher viewpoint is less affected by occlusion from the tall buildings, but a viewpoint higher than the standing height of the user is hard to achieve.

In this chapter, we propose to increase VR and AR scene exploration efficiency by enhancing the visualization with additional perspectives. The HMD shows to the

user a multiperspective image that captures the scene from multiple viewpoints, yet the image is non-redundant, with no parts of the scene being shown more than once, and continuous, with nearby 3D scene points projecting to nearby image locations. The ROIs occluded from the user's viewpoint are captured from secondary viewpoints and integrated into the original view. The additional perspectives are rendered with correct disparity so the "stereo" multiperspective HMD visualization provides appropriate depth cues to the user. We employ two types of multiperspective visualization, designed to alleviate occlusions in two scenarios.

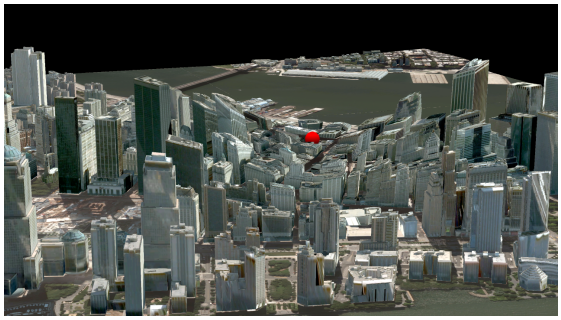
The first type of multiperspective visualization is designed to overcome occlusions in an urban VR scene (Fig. 3.1, top). The second type is designed for AR visualization of environments defined by cells connected by portals, such as a building interior with rooms and corridors connected by doors and intersections (Fig. 3.1, bottom). The visualization supports the simultaneous disocclusion of multiple ROIs (Fig. 3.2), as well as the integration of RGBD streams acquired with depth cameras (Fig. 3.3).

In order to quantify any VR and AR navigation benefits brought by the multiperspective visualization, we have conducted a randomized controlled user study in which the subjects were asked to perform four visual and navigational tasks in the environments shown in Fig. 3.1. Our multiperspective VR and AR reduced viewpoint translation by 45.2% and view direction rotation by 43.2%. We also refer the reader to the accompanying video. To the best of our knowledge, our work is the first investigation of multiperspective occlusion visualization in VR and AR HMD applications.

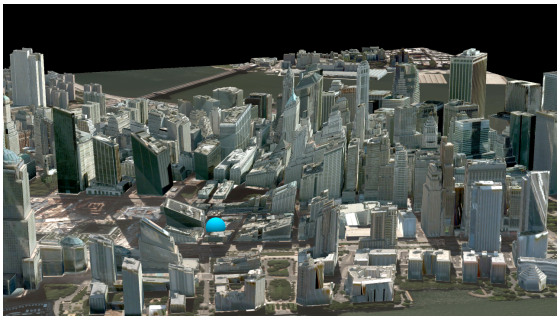
3.2 Prior work

In VR applications, viewpoint navigation in the virtual scene is performed by physical locomotion in a host real world scene. The quality of the navigation experience is essential to the success of the VR application. One challenge stems from a mismatch between the size of the physical world accessible to the user and where the

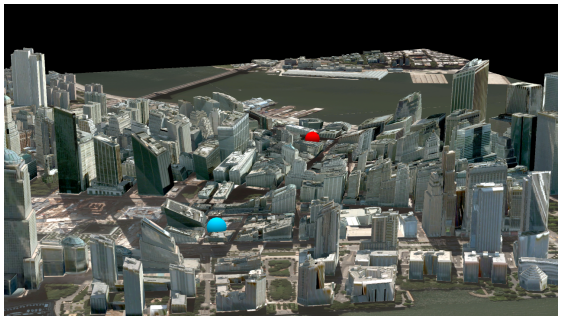
Fig. 3.1.: (Top) Multiperspective disocclusion in a VR exploration of an urban scene: artist rendition of a second-person view (left), conventional VR visualization (middle), and our multiperspective VR visualization (right). The user does not have a direct line of sight to the red sphere, which is occluded in the a conventional image; an additional perspective (green frustum) is used to route rays over the occluding buildings (dashed blue line), which disoccludes the red sphere in the resulting multiperspective image. (Bottom) Multiperspective disocclusion in an AR exploration of a real world indoor scene: second-person image (left), conventional image from user viewpoint (middle), and our AR multiperspective visualization (right). The left side corridor is disoccluded by inserting an additional perspective at the portal (red rectangle), which reveals the target (orange ghost).



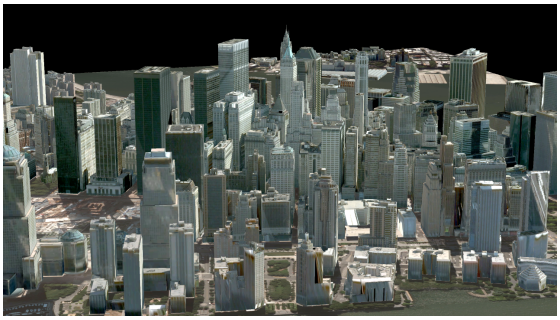
(a) Red sphere disoccluded



(b) Blue sphere disoccluded



(c) Both spheres disoccluded



(d) Conventional visualization

Fig. 3.2.: Simultaneous disocclusion of multiple ROIs through our multiperspective visualization.

Fig. 3.3.: Left: the RGB camera with depth sensor accessory captures a RGBD video stream that is converted to a 3D point cloud. Right: the point cloud, as seen from the secondary perspective, is integrated into the primary perspective in the multiperspective visualization.

user can be tracked, and the size of the virtual world that the user wants to explore. We discuss prior work that addresses this challenge in Section 3.2.1. A second challenge stems from the reduction in visualization efficiency caused by occlusions. In conventional VR visualization, the user can only see parts of the scene to which there is a direct line of sight. The user has to explore the scene sequentially, walking to reach viewpoints from where region of potential interest is visible. We discuss existing approaches for occlusion management in Section 3.2.2, and we compare our proposed technique to prior multiperspective techniques in Section 3.2.3.

3.2.1 Physical to virtual world mapping

The virtual scene is frequently more expansive than the available physical space. This limitation is worked around using in-place walking systems based on treadmills [40] or inertial sensing hardware [41]. These methods interpret a subset of the user's motion associated with walking, such as feet or head motion, to infer the desired locomotion. The drawback to in-place navigation systems is that the user does not change the viewpoint naturally by walking. Our system allows the user to walk freely, and, when desired, the user can save on the amount of walking by inspecting an occluded area with the help of an intuitively deployed secondary viewpoint. If the occluded area turns out to be a region of interest, the user can walk to it for further intuitive visualization.

When the physical space is less restrictive, locomotion by actual walking is found to be more intuitive for users [42, 43]. The problem of navigating a large virtual scene in a smaller physical space is well established. There exist methods such as redirection [44, 45], variable scene scaling [46], variable translation gain [47], and pose resetting [48]. Another approach is an optimized non-linear mapping from physical to virtual space [49]. Although effective at extending the virtual scene beyond the physically available space, these methods cause persistent discrepancies between perception and reality, of which the user can become aware, making the navigation less intuitive

[50, 51]. Our system does not attempt to hide the added supernatural visualization capability, which is deployed at the user's demand, with the secondary viewpoint visualization anchored in the familiar single-perspective visualization from the user's viewpoint. Through the use of the secondary viewpoint, our system provides a limited extension of the virtual world beyond the physical locations that the user can reach.

In AR applications, the graphical annotations must remain registered to the physical scene, therefore any navigation method that does not preserve the identity mapping between virtual and physical spaces is ill-suited to AR. Because the VR methods mentioned improve navigation efficiency by diverging the virtual viewpoint from the user's physical location, they are not applicable to AR.

3.2.2 Occlusion management

Occlusion management is a promising approach to improving the efficiency of viewpoint navigation in both VR and AR, while retaining navigation intuitiveness. Occlusions reduce visualization effectiveness when the line of sight to the ROI is blocked by opaque objects. Conventional approaches to occlusion management fall into categories of X-ray, cutaway, and explosion methods. Multiperspective approaches for managing occlusions are discussed separately, in Section 3.2.3.

In X-ray visualization, the occluder is rendered semi-transparently so that occluded scene segments are visible through the occluder [52]. However, this visualization technique violates pictorial depth cues because the background is blended with the foreground. To alleviate this problem, the *ghosting* approach emphasizes foreground information at the edges [53], or other salient features [54, 55], by overlaying them as a ghost image. While adding a single processed foreground layer restores missing depth cues [56], scaling with occluder complexity, which requires visualization of multiple transparent layers, remains challenging. In addition, the compromise must still be made between conveying foreground information and limiting visual complexity, due to the overlapping image space position of the occluder and the ROI.

In cutaway visualization, the occluder is completely removed instead of being visualized transparently [35,57]. The removal of the occluder leaves a hole for the line of sight to pass through and reach the ROI. Cutaway visualization maintains correct depth cues, but extra geometry must be generated for cut surfaces surrounding the hole, and little information about the removed occluders is conveyed.

Explosion methods segment the scene and translates the segments radially away from the ROI to reduce occlusion. The method is widely employed in computer aided design applications where the constituent parts are naturally separable by clear boundaries, in accordance to assembly sequences [34]. Volumetric datasets support explosion visualization when scene segments are annotated [33]. In AR applications, explosion visualization is possible when there exist synthetic 3D models corresponding to the physical objects [58]. In this case, the 3D models are required to visualize scene segments from alternative perspectives as they are translated due to the explosion. Compared to X-ray visualization, which increases visual complexity, and cutaway visualization, which omits foreground information, the explosion approach is able to present the foreground without interfering with the visualization of the ROI. However, explosion methods artificially fragment scene geometry which causes discontinuities across scene segments. Furthermore, the translation of scene segments hinder the user's spatial awareness, which is undesirable in VR and AR applications. Hybrid scene deformation methods seek to minimize disturbance to the scene geometry by combining multiple operations such as scene segment translation, scaling, and viewpoint shifting [59]. However, this approach relies on extensive scene preprocessing to identify scene segments, and optimization of cost functions tailored to these scene segments. The viewpoint shifting operation also interferes with the navigation in VR and AR.

Our method takes the multiperspective visualization approach, and we discuss prior work in this category in detail in the next subsection. X-ray, cutaway, and explosion visualization have the advantage of intuitiveness, as they can be seen as applying a familiar change to the real world: the occluding layer is built out of trans-

parent material, a hole is cut into the occluding layer, or the scene is disassembled into individual parts. There is no familiar real world manipulation that achieves a similar effect to multiperspective visualization. However, multiperspective visualization is a powerful occlusion management technique with unique strengths.

3.2.3 Multiperspective visualization

Multiperspective visualization was first used in the visual arts, such as by Picasso in his cubist paintings. The single viewpoint constraint is relaxed in favor of more expressive images containing multiple integrated views. In imaging research, the study of camera models progressed beyond the traditional pinhole camera to novel cameras such as the push broom [60], the multiple center-of-projection [19], and the general linear camera models [61]. The goal is comprehensive acquisition of real world scenes with powerful imaging systems that overcome occlusions by capturing rays from multiple viewpoints.

In desktop visualization, multiperspective cameras were developed to increase the information bandwidth of images by alleviating occlusions. Occlusion cameras [23] are a class of multiperspective cameras that generalize the viewpoint to a view region by routing sampling rays around occluders. The curved ray camera routes rays around occlusions through multiple sequential viewpoints with C^1 continuity, though it does not support branching to multiple viewpoints in parallel [29]. The graph camera [24] generates a continuous and non-redundant multiperspective image that integrates multiple disparate viewpoints using frustum bending, splitting, and merging operations. The graph camera is literally a graph of conventional planar pinhole cameras. The node cameras were subsequently upgraded to general linear cameras to achieve multiperspective focus+context visualization: ROIs are shown from secondary viewpoints, and the perspective reverts to the main viewpoint outside the ROIs [1].

Several prior multiperspective visualization efforts target specifically urban and terrain scenes and opt for the approach of deforming distant, occluded geometry upward, while pushing near, occluding geometry downward [62–64]. One system improves spatial awareness through disocclusion in large-scale scenes, by providing the user with a choice of a large number of video feeds acquired from multiple vantage points [62]. The user selects the desired first person view from a third person view of the available feeds, and then resorts to a two-viewpoint multiperspective visualization to disocclude in the first person view. Compared with these techniques, our method has the advantage of not distorting the ground plane, which prevents user disorientation. Furthermore, our technique visualizes the disoccluded ROI exactly where it would be seen in the absence of occluders, which enables building a mental model of the target location. Benefiting from the stable ground geometry, our visualization supports an intuitive user interface where a ROI is directly selected by the user's gaze towards the ground plane, as opposed to manually inputting deformation parameters [62]. When multiple ROIs are identified, our visualization deploys multiple secondary views to disocclude individual ROIs with localized deformation of the geometry, whereas prior work deforms large terrain partitions without low-level granularity of the disocclusion effect [62–64].

Our visualization is able to augment the main physical view with secondary views in AR. Prior work displays secondary views by inserting virtual billboards into the scene [62,65]. This has the advantage of allowing the user to teleport to distant viewpoints [62], whereas our method restricts the user to portals within sight. However, the billboards displaying the secondary viewpoints in the overview suffer from visualization discontinuity. Furthermore, the visualization based on monoscopic video lacks depth perception, which our method achieves by rendering the geometric model for each eye. Other prior work inserts deformed 3D geometry into the scene [66], but there is significant discontinuity between the deformed geometry and the physical world. In our AR visualization, secondary views are inserted as 3D geometry that seamlessly integrates with the un-deformed physical geometry.

We choose to build our multiperspective visualizations based on the graph camera [24] and on the multiperspective focus+context [1] frameworks. We extend these prior multiperspective frameworks to achieve an intuitive deployment of the additional viewpoints based on head-tracking, and to achieve stereoscopic multiperspective rendering as needed for the HMD. We measure the navigation efficiency increases in a controlled user study with two VR and two AR tasks.

3.3 Multiperspective VR visualization in urban scenes

In the urban VR scene, ROIs at street level are occluded by tall buildings. Such ROIs become disoccluded from an overhead perspective that has a viewpoint above the ROI and a downward view direction. When the location of the ROI is not known, as is the case for the tracking and matching tasks considered in this chapter, we define the overhead perspective interactively under the assumption that the target is at the street-level point of intersection between the user's current view direction and the ground plane. Once the secondary, overhead perspective is defined, the scene is rendered with the resulting multiperspective camera model which integrates the secondary perspective seamlessly into the primary, user perspective. The resulting multiperspective image disoccludes the street-level look-at point without compromising the user's spatial awareness.

3.3.1 Interactive construction of secondary perspective

The user attempts to locate the ROI by scanning the urban model. The goal is to let the user see at street level, free of occlusions. The secondary perspective is constructed to provide an occlusion-free visualization of a 3D focus point defined as the intersection between the user's view direction and the ground plane. To aid the user in selecting the street-level point that should be disoccluded, a cursor is displayed at the focus point. The secondary viewpoint is placed vertically above the focus point at a height equal to the distance, r, from the primary viewpoint to the focus point

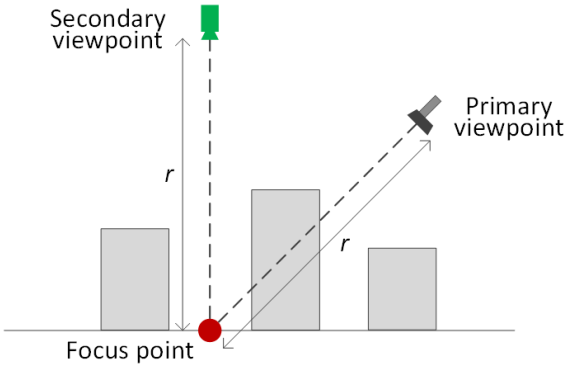


Fig. 3.4.: Secondary viewpoint placement

(Fig. 3.4). The secondary perspective looks directly downwards and avoids occlusion from nearby tall buildings. The secondary perspective is rendered with the same field of view as the primary perspective, so, since the distance to the focus point is in the same in both perspectives, the disoccluded ROI will be shown in the multiperspective image at the same scale at which it would be seen in the primary perspective in the absence of occlusions. We use the same secondary viewpoint for both the left and the right user's eyes, which we compute by defining the primary viewpoint as the midpoint of the segment between the two eyes.

3.3.2 Secondary view integration

The integration of the secondary view into the user's primary view of the scene has to achieve three goals: (1) it has to give preference to the secondary view around the ROI, in order to achieve the desired disocclusion effect, (2) it has to give preference to the primary view away from the ROI, in order to help the user remain aware of their position and orientation in the scene, (3) and it has to display the street-level ROI where the user would see it in the absence of occluders, for the user to get an accurate sense of where the ROI is located in the scene. We achieve the third goal by "anchoring" the ground plane, i.e. by enforcing that the ground plane is not distorted by the multiperspective visualization.

We render the multiperspective image by transforming the scene vertices with a three step operation. The results of the individual transformation steps are shown in Fig. 3.6. The intermediate steps are only shown here for illustration purposes; in our implementation, the vertices are projected directly from model space to the multiperspective image. In the first transformation step, a scene vertex u is transformed to the local coordinates v_1 and v_2 of the primary and secondary perspectives by multiplication with their respective transformation matrices $\mathbf{V}_1$ and $\mathbf{V}_2$. All matrices and vectors are in standard homogeneous coordinates.

$$\begin{aligned} v_1 &= \mathbf{V}_1 \times u \\ v_2 &= \mathbf{V}_2 \times u \end{aligned} \tag{3.1}$$

The second step anchors the transformed vertex in the secondary perspective to the ground plane in the primary perspective by offsetting v_2 by a displacement vector d.

$$\begin{aligned} v_2' &= v_2 + d \\ &= v_2 + (\mathbf{V}_1 \times u_{ground} - \mathbf{V}_2 \times u_{ground}) \end{aligned} \tag{3.2}$$

The 3D point u_{ground} is the ground plane projection of scene vertex u. For any vertex u on the ground plane where $u = u_{ground}$, the transformed vertex v_2' is equivalent to transforming u with $\mathbf{V}_1$, ignoring $\mathbf{V}_2$, and Eq. 3.2 becomes

$$\begin{aligned} v_2' &= v_2 + d \\ &= \mathbf{V}_2 \times u_{ground} + (\mathbf{V}_1 \times u_{ground} - \mathbf{V}_2 \times u_{ground}) \\ &= \mathbf{V}_1 \times u_{ground} \\ &= \mathbf{V}_1 \times u = v_1 \end{aligned} \tag{3.3}$$

Thus the resulting multiperspective image shows the ground plane the same way the primary perspective does.

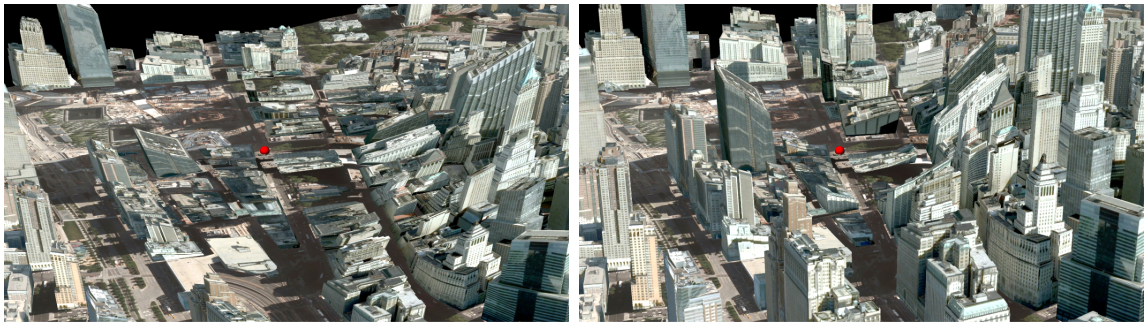


Fig. 3.5.: The amount of screen space taken up by the secondary perspective image depends on the width σ of the Gaussian G. Left: $\sigma = 0.8$; right: $\sigma = 0.4$

In the third and final step, the transformation of the vertex u is finalized as a blend between the anchored transformation and the the primary perspective transformation.

$$v = \mathrm{G}(r, \sigma)v_2' + (1 - \mathrm{G}(r, \sigma))v_1 \tag{3.4}$$

The blend weights are given by a Gaussian G centered at the center of the secondary perspective image. The distance r is computed in the secondary perspective image as the distance between the center of the image and the projection of the vertex. The root-mean-square width σ is set according to the application (Fig. 3.5). Once the vertex is transformed with Eq. 3.4, the transformed vertex is then multiplied with a conventional projection matrix, that is the same for both the primary and secondary perspectives.

In the case of multiple ROIs, each ROI defines its own focus point and secondary perspective. Eq. 3.4 is modified such that the secondary perspectives are first integrated with one another (Eq. 3.5), and then the result is integrated with the primary perspective (Eq. 3.6).

$$\bar{v} = \frac{\sum_{i>1} \mathrm{G}(r_i, \sigma)v_i'}{\sum_{i>1} \mathrm{G}(r_i, \sigma)} \tag{3.5}$$

$$v = \mathrm{G}_{max}\bar{v} + (1 - \mathrm{G}_{max})v_1 \tag{3.6}$$

v_i' is the vertex transformed according to secondary perspective i, and r_i is the screen space distance to the center of the image of secondary perspective i. G_{max} is the largest of the weights $G(r_i, \sigma)$. This gives preference to the secondary perspective transformation over the primary perspective transformation if a vertex is close to the center of any of the multiple ROIs. Consider a case with 3 ROIs. A vertex at the center of the first ROI and peripheral to the other two ROIs, i.e. which has transformation weights approximately equal to (1, 0, 0), should be transformed according to the secondary perspective of the first ROI. Using the average of the weights $G(r_i, \sigma)$ in Eq. 3.6 would incorrectly give the first ROI transformation only a 33% weight.

The multiple secondary perspectives do not have to be disjoint. In the case when two dynamic ROIs become close or coincident, their respective secondary viewpoints also become close or coincident while sharing the identical downward view direction. Therefore, as two ROIs approach each other, their associated secondary perspectives smoothly converge, and the multiperspective visualization remains continuous.

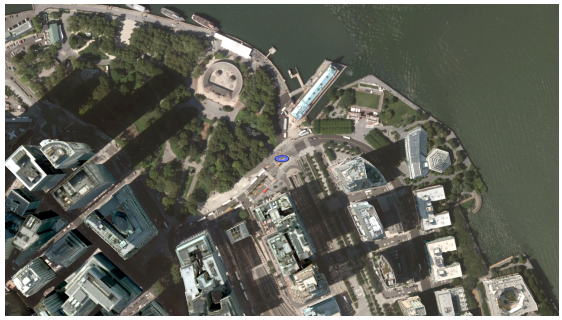
3.3.3 Stereoscopic rendering

The HMD conveys depth cues through a stereoscopic image rendered from the left and the right eye viewpoints, separated by the interpupillary distance. To support stereoscopic rendering in our multiperspective visualization, each vertex v in the coordinates of the monocular primary perspective is transformed to the coordinates of the left and the right perspectives, v_L and v_R.

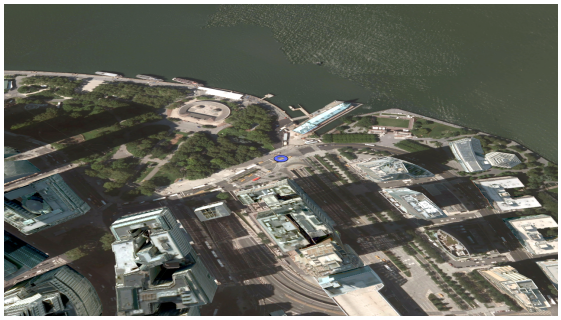
$$\begin{aligned} v_L &= \mathbf{V}_L \mathbf{V}_1^{-1} \times v \\ v_R &= \mathbf{V}_R \mathbf{V}_1^{-1} \times v, \end{aligned} \tag{3.7}$$

where $\mathbf{V}_L$ and $\mathbf{V}_R$ are transformation matrices of the left and the right perspectives. The resulting multiperspective visualization has the correct disparity over the primary and secondary perspectives, conveying appropriate depth cues to the user.

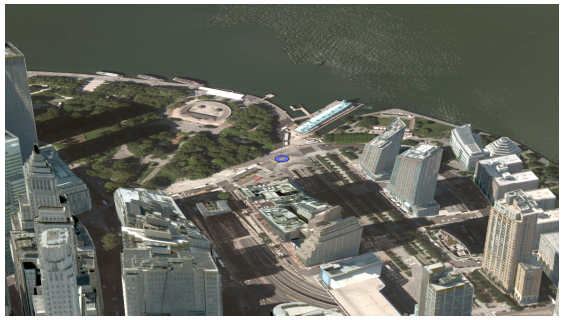
(a) Primary perspective



(b) Secondary perspective



(c) Anchored secondary perspective



(d) Integrated perspectives

Fig. 3.6.: Illustration of the multiperspective transformation. Both the primary (a) and secondary (b) perspectives are centered on the ground plane focus point, which is occluded in the primary perspective and disoccluded in the secondary perspective. The secondary perspective is first modified (c) for the ground plane to appear the same way it would appear in the primary perspective in the absence of occlusions. The final multiperspective image (d) transitions smoothly from the primary perspective, at the periphery, to the secondary perspective, at the center.

3.4 Multiperspective indoor AR visualization

Efficiency of exploration of a real-world indoor scene is limited by the tight turns at corridor intersections. In order to inspect the side corridors, the user has to walk to each intersection. Our AR multiperspective visualization leverages additional views of the corridors acquired by cameras placed at the intersections to present a comprehensive visualization to the user, who can see down a side corridor without having to walk up to the intersection. The secondary perspective down the corridor is seamlessly integrated with the user's primary perspective. In this section, we first describe the scene acquisition from additional perspectives. Then we describe the selection and integration of secondary views tailored to the user's current viewpoint.

3.4.1 Acquisition

Unlike in the case of VR synthetic scenes, the AR context requires capturing the additional perspectives with physical cameras. Furthermore, the multiperspective visualization requires rendering the additional perspectives from novel viewpoints, so a 2D image is not sufficient, and a geometric model is needed. In the indoor context, the building geometry (i.e. walls, ceiling, floors) is fixed and we model it with a simple geometric model projectively texture mapped with a photograph. For the controlled experiments described in this chapter, the content of the corridor is synthetic (i.e. we use a conventional computer graphics model of the ghost seen in Fig. 3.1, bottom. Dynamic real-world geometry is acquired with a RGBD camera, whose frames are reprojected to the desired viewpoint as a cloud of 3D points with color (Fig. 3.3).

3.4.2 Selection and integration of the secondary view

When the user sees an intersection, the rectangular portals leading to side corridors are highlighted with a red wireframe. If the user centers and holds the view on a portal, a secondary perspective swings into place, revealing what is visible through

the portal (Fig. 3.1, bottom). The deviation from the conventional view of the scene is confined to the area of the portal. The portal frame acts like a hinge connecting the primary and secondary perspectives with C_0 continuity.

When a portal is activated, the scene geometry beyond that portal is visualized by rotating the scene vertices into view. Vertex transformation proceeds similarly to that of VR visualization, where the first step is to transform each scene vertex to coordinates of the primary and secondary perspectives, and the second step is to anchor the geometry to the portal plane for the primary perspective (Fig. 3.7). However, there is no need for blending between secondary and primary perspectives because they integrate at the boundary of the portal.

We wish to restrict virtual scene distortion to the 2D horizontal plane, so the primary and secondary perspectives are first restricted to be horizontal and at the same height. This restriction is relaxed at the final step of the rendering pipeline, where the full unrestricted primary perspective is used for rendering the final image.

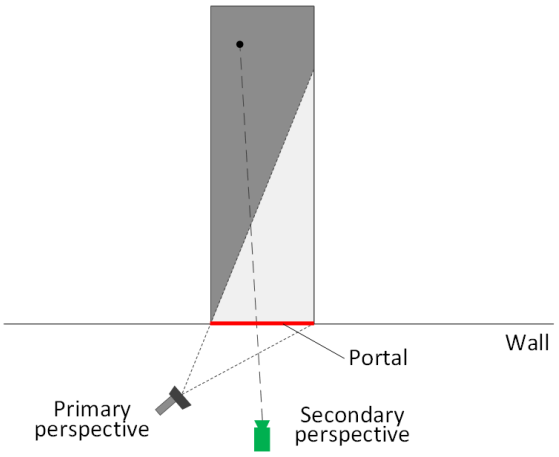
In the first step, the primary perspective is constructed with the viewpoint at the position of the HMD, and the view direction is towards the center of the portal. Each scene vertex u beyond the portal plane is transformed to both the primary and secondary perspectives using their respective transformation matrices, V_1 and V_2.

$$\begin{aligned} v_1 &= \mathbf{V}_1 \times u \\ v_2 &= \mathbf{V}_2 \times u \end{aligned} \tag{3.8}$$

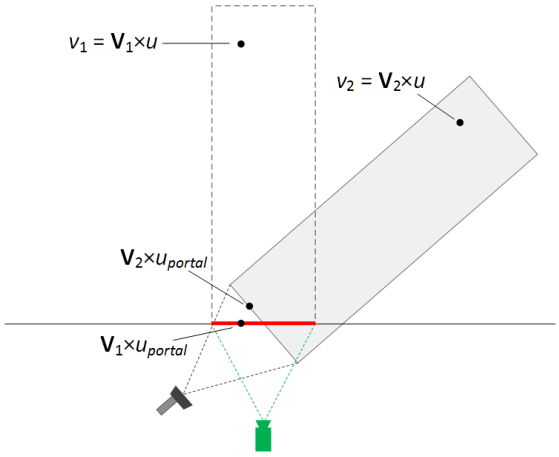
The second step anchors the vertex v_2 in the coordinates of the secondary perspective to the portal plane by offsetting v_2 by a displacement vector d.

$$\begin{aligned} v_2' &= v_2 + d \\ &= v_2 + (\mathbf{V}_1 \times u_{portal} - \mathbf{V}_2 \times u_{portal}), \end{aligned} \tag{3.9}$$

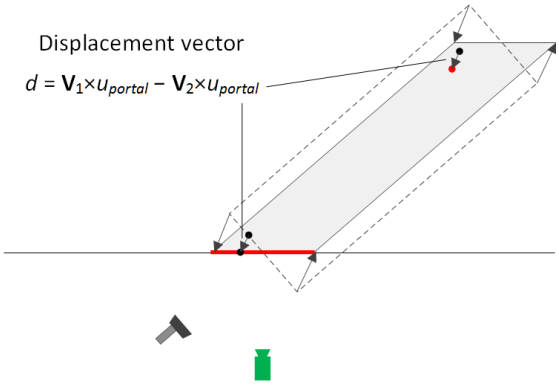
where u_{portal} is the projection of vertex u to the plane of the portal.



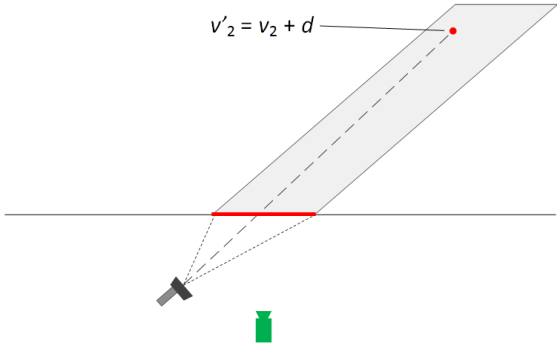
(a) Un-deformed geometry



(b) Step 1: Transforming to primary and secondary perspectives



(c) Step 2: Anchoring to to portal plane



(d) Deformed geometry

Fig. 3.7.: (a): the un-deformed geometry, where the target is occluded along with the shadowed parts of the corridor, although they are visible from the secondary viewpoint. (b): the geometry in coordinates of both primary and secondary perspectives. (c): the geometry anchored to the portal plane. (d): the deformed geometry, where the target and all of the corridor become visible from the primary viewpoint. The depth beyond the portal is preserved for each vertex.

For any vertex u on the portal plane where $u = u_{portal}$, the transformed vertex v_2' is identical to v_1, and Eq. 3.9 becomes

$$\begin{aligned} v_2' &= v_2 + d \\ &= \mathbf{V}_2 \times u_{portal} + (\mathbf{V}_1 \times u_{portal} - \mathbf{V}_2 \times u_{portal}) \\ &= \mathbf{V}_1 \times u_{portal} \\ &= \mathbf{V}_1 \times u = v_1 \end{aligned} \tag{3.10}$$

Thus the resulting multiperspective image shows the portal plane without any deformation, and the secondary perspective is anchored to the primary perspective on the portal plane.

A deformed side corridor is shown in Fig. 3.7. The target, represented by a red circle, is occluded from the primary perspective, but is visible from the secondary perspective. The target becomes visible when the scene geometry is deformed such that the corridor beyond the portal is aligned with the primary perspective. Fig. 3.1 (right) shows the rendered AR result.

Multiple secondary perspectives can be active simultaneously. Geometry visible through each portal is deformed separately according to each secondary perspective. No blending between secondary perspectives is necessary because the portals are disjoint, and the visualization of the deformed geometry is confined to the area of the portals.

With modified scene geometry, stereoscopic rendering proceeds straightforwardly by projecting scene vertices separately to the left and the right eye coordinates using their respective transformation matrices V_L and V_R, as described in Section 3.3.3.

3.5 User study

We have conducted a controlled randomized user study to evaluate the effectiveness of our VR and AR multiperspective visualization methods. In this first study we

evaluate our method against conventional planar pinhole camera visualization, which is used in the overwhelming majority of VR and AR applications. We implemented our visualization methods for the urban VR and indoor AR scenes using an HMD with SLAM user tracking (i.e. the Microsoft HoloLens). The urban VR scene is a photo-textured model of Manhattan that is mapped to an empty floor space of $4m \times 5m$ (Fig. 3.1, top). The indoor AR scene covers a $10m \times 15m$ section of the floor plan of our office building (Fig. 3.1, bottom).

3.5.1 Subjects

We recruited a total of 16 subjects for our study, 12 male and 4 female. The subjects were undergraduate and graduate students between the age of 19 and 42. The subjects were randomly assigned to equal-sized control and experimental groups of 8 subjects each. The subjects assigned to the control group used a conventional visualization that showed only the primary perspective. The other subjects, assigned to the experimental group, used the multiperspective visualization. Each subject performed all four tasks directly, without training and without familiarization with the multiperspective visualization and with the user interface. The control group subjects wore the HMD even for the AR tasks, when the portals were highlighted with the red wireframe, but without the additional perspectives. In addition to age, the demographic information collected from the subjects also covered prior AR and VR experience. From the 16 subjects, 8 had prior experience with VR applications, and 6 had prior experience with AR applications.

3.5.2 Tasks and data collection

Each subject performed four tasks. Two of the tasks are performed in the urban VR scene, and the other two are performed in the indoor AR scene. The tasks require subjects to gain and maintain sight of static and dynamic synthetic objects placed in the scenes. The control and experiment configurations of the tasks are identical except

for the visualization method employed. The user's head position and orientation are tracked and saved for data analysis.

Target tracking task (VR1) In the first urban VR task the subject is asked to keep a target in view as the target moves on the streets of the city. The target moves for 38s, after which the task is complete.

Pair matching task (VR2) In the second urban VR task the subject is asked to find pairs of spheres with the same color pattern. There are six pairs of spheres placed at street level and scattered throughout the scene. The subject selects a sphere by centering the view on it and by clicking a hand-held wireless mouse. In the experimental group, the selected sphere is marked as a ROI and is kept disoccluded even when the subject changes focus. The subject clicks are recorded for analysis of the rate of matching error. When two matching spheres are selected, the spheres are removed from the scene, and the task is completed when no sphere remains.

Search task (AR1) In the first AR task, the subject is asked to explore the corridors of the scene in search of stationary targets, implemented as computer graphics "ghosts" (Fig. 3.1.) Only one target is available at one time. A target is found and removed when the user moves within 2m of it. Once a target is found, the next target is placed in the scene, which forces the user to revisit the scene in search of the new target. The task is completed after four targets are found. The target locations are the same for the control and experiment conditions.

Ambush task (AR2) The second AR task is similar to the first one, except that now the targets move. When the subject is in the direct line of sight of the target, the target moves away from the subject. This requires the subject to "ambush" the target, by waiting around a corner for a target to move into the intersection where it is "captured". The targets move with the speed of 1.2m/s, so the subject can also run after the evading target to catch up with it. When a target is captured, the next target is spawned in a part of the scene far away from the current position of the subject. The subject is informed of the target's evasive motion strategy, but is not advised of a counter strategy.

3.5.3 Results and discussion

We analyze scene navigation performance using several metrics extracted from the subject pose traces recorded during our experiments. We detect and measure benefits of the multiperspective visualization compared to the conventional visualization in two ways: using Cohen's effect size d [67], and using a two-sample t-test. Cohen's d is calculated from measurements performed on the control and experimental groups, and its value is the difference between the mean values divided by the pooled standard deviation. Effect sizes are conventionally qualified as "small", "medium", and "large" for the cases where $d > 0.2$, $d > 0.5$, and $d > 0.8$ respectively. Due to the variation of effect sizes observed in our metrics, the qualifiers for effect sizes are expanded to include "very small", "very large", and "huge" for $d < 0.01$, $d > 1.2$, and $d > 2.0$ respectively [68]. We investigate the statistical significance of the measured improvements using a two-sample t-test, reporting the probability p (i.e. p-value) for the measured improvements to be due to chance. We first discuss metrics used for all four tasks, and then we discuss metrics specific to individual tasks.

Viewpoint translation

One metric common to all tasks is subject viewpoint translation. Our visualization brings in additional perspectives that are invoked and examined with intuitive head motions and without requiring locomotion to assume their corresponding viewpoints. Therefore, for all tasks, we expect a reduction in the total distance traveled by the subjects in the experimental group compared to those in the control group. As shown in Table 3.1, the total distance traveled by the subjects in the experimental group is significantly shorter. For example, for the VR1 task, 68% of the control group subjects traveled between $13.7m - 3.4m$ and $13.7m + 3.4m$ (i.e. one standard deviation). The difference between the averages is 8.3m, and Cohen's d is 2.3, which corresponds to a huge effect size. The largest benefit of multiperspective visualization is measured for task VR2, where the experimental group subjects learned how to

Table 3.1.: Average distance traveled per subject, in meters.

task	control	experiment	difference	p	d	effect size
VR1	13.7 ± 3.4	5.4 ± 3.8	8.3	< 0.01	2.3	huge
VR2	91.7 ± 47.0	8.2 ± 7.3	83.4	< 0.01	2.5	huge
AR1	99.1 ± 3.0	84.8 ± 12.3	14.4	< 0.01	1.6	very large
AR2	109.3 ± 16.9	93.3 ± 17.6	16.0	0.04	0.9	large

search for targets by staying in place, and where the control group subjects move repeatedly back and forth as they forget the colors of the targets. The smallest, but still large, benefit of multiperspective visualization is measured for task AR2, where the targets are spawned as far away as possible from the subject. For the VR tasks, the improvement brought by multiperspective visualization is statistically significant with p-values below 0.01. Due to the variable nature of target spawn locations, measurements for task AR2 also suffer slightly lower statistical significance (p-value of 0.04). Overall, the large effect sizes measured for this and other metrics (Tables 3.2, 3.3, and 3.4) justify the number of subjects used in the control and experimental groups, i.e. eight and eight.

Fig. 3.8 shows the actual trajectories of the control and experimental group subjects who moved the least while completing the urban VR tasks. The control group subject covers a significant part of the scene, whereas the experimental group subject stays within 0.2m of the starting position.

View direction rotation

The second metric common to all tasks is subject view direction rotation. The multiperspective visualization performs automatically most of the view direction rotation needed for the user to gain line of sight to the target. Therefore, for all tasks, we also expect a reduction in the total view direction rotation performed by the subjects in the experimental group. As shown in Table 3.2, the subjects in the experimental group rotated their view direction significantly less. All p-values are less than 0.05.

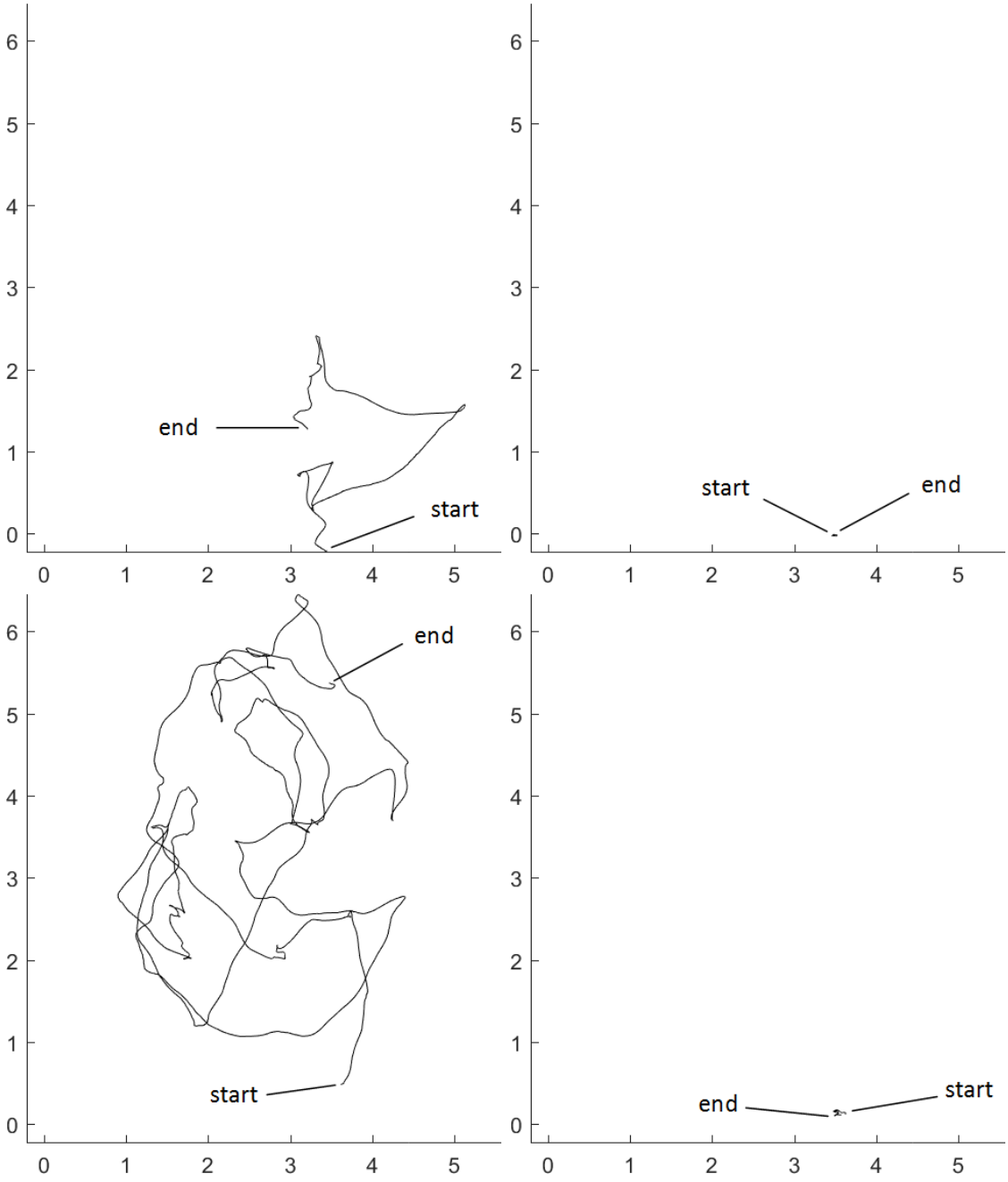


Fig. 3.8.: User viewpoint 2D trajectory visualizations for the urban VR tracking (top) and pair matching (bottom) tasks. When using a conventional visualization (left), the user moves over considerable distances. When using the multiperspective visualization (right), the user can complete the task by standing in one place.

Table 3.2.: Total head rotation, in hundreds of degrees.

task	control	experiment	difference	p	d	effect size
VR1	7.8 ± 1.3	4.4 ± 2.4	3.3	< 0.01	1.8	very large
VR2	66.0 ± 22.5	10.0 ± 4.8	56.0	< 0.01	3.4	huge
AR1	41.6 ± 6.9	32.8 ± 10.8	8.7	0.04	1.0	large
AR2	43.9 ± 8.7	33.7 ± 6.8	10.1	0.01	1.3	very large

Table 3.3.: Average head downward rotation, in degrees.

task	control	experiment	difference	p	d	effect size
VR1	57.9 ± 14.3	40.9 ± 14.2	17.0	0.02	1.2	very large
VR2	48.5 ± 4.1	28.4 ± 5.2	20.0	< 0.01	4.3	huge

For the target tracking (VR1) and pair matching (VR2) tasks, where subjects tilt their head downwards to look at targets at street level, we analyzed the average downward head rotation (Table 3.3). The subjects in the experimental group required significantly less downward head rotation ($p < 0.02$), which is an important improvement. When the subject looks down close their feet, the HMD's limited field of view covers only a small part of the scene, which requires moving a lot to search for and to track the target. In addition, seeing only a small part of the scene reduces spatial awareness, which further complicates the tasks. When the view direction is less tilted, the scene is farther away and more of it is visible, which improves task completion efficiency. Finally, viewing the scene with a downward titled view direction is uncomfortable since it leads to an imbalanced distribution of the HMD's weight, as spontaneously reported by two of the subjects.

Task completion time

Table 3.4 gives completion times for the tasks, with the exception of VR1 where the task is complete at the end of the fixed 38 second target trajectory. In the pair matching task (VR2), subjects complete the task in significantly shorter time in the experimental group ($p < 0.01$). This is expected because the subject is able to simultaneously examine two spheres, when one of them is marked as ROI and kept disoccluded, and the other is found and disoccluded by focusing the view on it. In contrast, the subject in the control group is only able to examine the spheres sequentially, with significant physical motion in between. In the search task (AR1), the subject in the experimental group benefits from the multiperspective visualization which shows multiple additional perspectives simultaneously without requiring phys-

Table 3.4.: Task completion time, in seconds.

task	control	experiment	difference	p	d	effect size
VR2	215.5 ± 86.4	57.8 ± 26.5	157.7	< 0.01	2.5	huge
AR1	91.5 ± 8.0	69.0 ± 19.5	22.5	< 0.01	1.5	very large
AR2	75.1 ± 24.9	73.6 ± 19.3	1.5	0.45	0.1	very small

ical locomotion, while the subject in the control group must translate their viewpoint physically, and is only able to examine side corridors sequentially. Therefore, task AR1 is performed significantly faster in the experimental group than in the control group ($p = 0.01$). In the ambush task (AR2), however, subjects in the experiment and control groups take about the same time to complete the task. The subject in the experimental group is able to leverage the multiperspective visualization to quickly locate the mobile target, but may choose to spend time to wait to ambush. This strategy results in overall less physical locomotion for the experimental group, but not in faster task completion.

Task-dependent metrics

For the target tracking task (VR1), we measured for what percentage of the time the subject succeeded at keeping the target in sight. The target sphere is considered visible if any part of it is in the subject's view. In the experimental group, the subjects achieved 97.5% of target visibility compared to the control group, which achieved only 74.1% of target visibility. In other words, even by walking continually, the subject cannot always keep the target in sight. When the target makes a turn, the user has to translate significantly to realign with the street on which the target moves, which takes time. The improvement is 23.3 percentage points with a Cohen's d of 1.4, which corresponds to a very large effect size. This effect is statistically significant ($p = 0.01$).

For the pair matching task (VR2), the subject is asked to select two spheres of the same color in succession. If the second sphere selected is of a different color, then this pair of selections is considered a matching error. The subjects in the experimental

group incurred 1.6 erroneous selections on average, while the subjects in the control group incurred a higher rate of 2.5 erroneous selections on average. The improvement is 0.9 selections with a Cohen's d of 0.4, a small effect size. The effect is not statistically significant ($p = 0.22$).

Effect of subject prior AR/VR experience on task performance

Due to the absence of familiarization periods allowed before each task, the subjects' prior experiences with VR and AR applications were analyzed for any possible effect on task performance. From the eight subjects with prior VR experience, four were assigned to the control group and four to the experiment group, for each of the tasks VR1 and VR2. For AR1, from the six subjects with prior AR experience, two were assigned to the control group and four were assigned to the experiment group. For AR2, three of the AR experienced subjects were assigned to each of the two groups. Overall, there were no consistent differences between subjects with and without prior experience within single groups, and any difference found was statistically insignificant. Due to the small number of subjects with experience in a group, a dedicated user study is needed to investigate fully the effect of prior experiences when using our visualization methods.

3.6 Conclusions and future work

We have presented novel multiperspective visualizations to improve navigation efficiency in AR and VR. Our visualization techniques seamlessly integrate perspectives from multiple secondary viewpoints into the main user perspective. The secondary perspectives are selected by the user intuitively, with minimal or no interface manipulation. The output supports stereoscopic displays by rendering the scene with correct depth cues. We demonstrate the effectiveness of our visualization techniques in a user study where subjects were asked to accomplish tasks in AR and VR using an HMD with tracked pose. Based on the study, we report significant improvement in

navigation efficiency while using multiperspective visualization compared with using conventional visualization.

There exist limitations to our visualization techniques. The urban VR visualization assumes that any ROI at street level can be disoccluded from an overhead perspective. This assumption does not hold, for example, when the target object hides underneath a bridge. It is possible to include additional panning control to the construction of secondary views, but this increases the complexity of the user interface. If the target is unreachable, e.g., it hides inside an enclosed space, then even the addition of panning control cannot bring it into view, and we must resort to X-ray or cutaway visualizations. The indoor AR visualization seamlessly swings side corridors into view, but the deformation operation can only guarantee continuity between real and virtual scene parts joined at a planar portal. Supporting non-planar portals is possible at the cost of visual complexity.

Fig. 3.9 compares our multiperspective visualization approach to handling occlusions to X-ray visualization, for both the urban VR and the indoor AR scenes. For the VR scene, the X-ray visualization does not convey the location of the target (red sphere) with respect to the network of streets. The multiperspective visualization anchors the target at the correct street location, and, unlike a simple overhead view, does also convey the height of the nearby buildings. For the AR scene, the X-ray visualization covers a significant fraction of the nearby occluding wall with the side corridor, which obscures parts of the scene close to the user. Our multiperspective visualization does not blend mismatching colors, which maintains the clarity of the visualization, and confines the depiction of the side corridor to the user's view of the portal. Overall, our visualization presents a different approach to the problem of missing depth cues suffered by X-ray views.

Our user study relies on conventional visualization for the control group. The user manages occlusions by physically navigating the viewpoint to locations from where there is line of sight to potential ROIs. This first study provides a useful baseline for our visualization approach. Future studies could attempt to compare

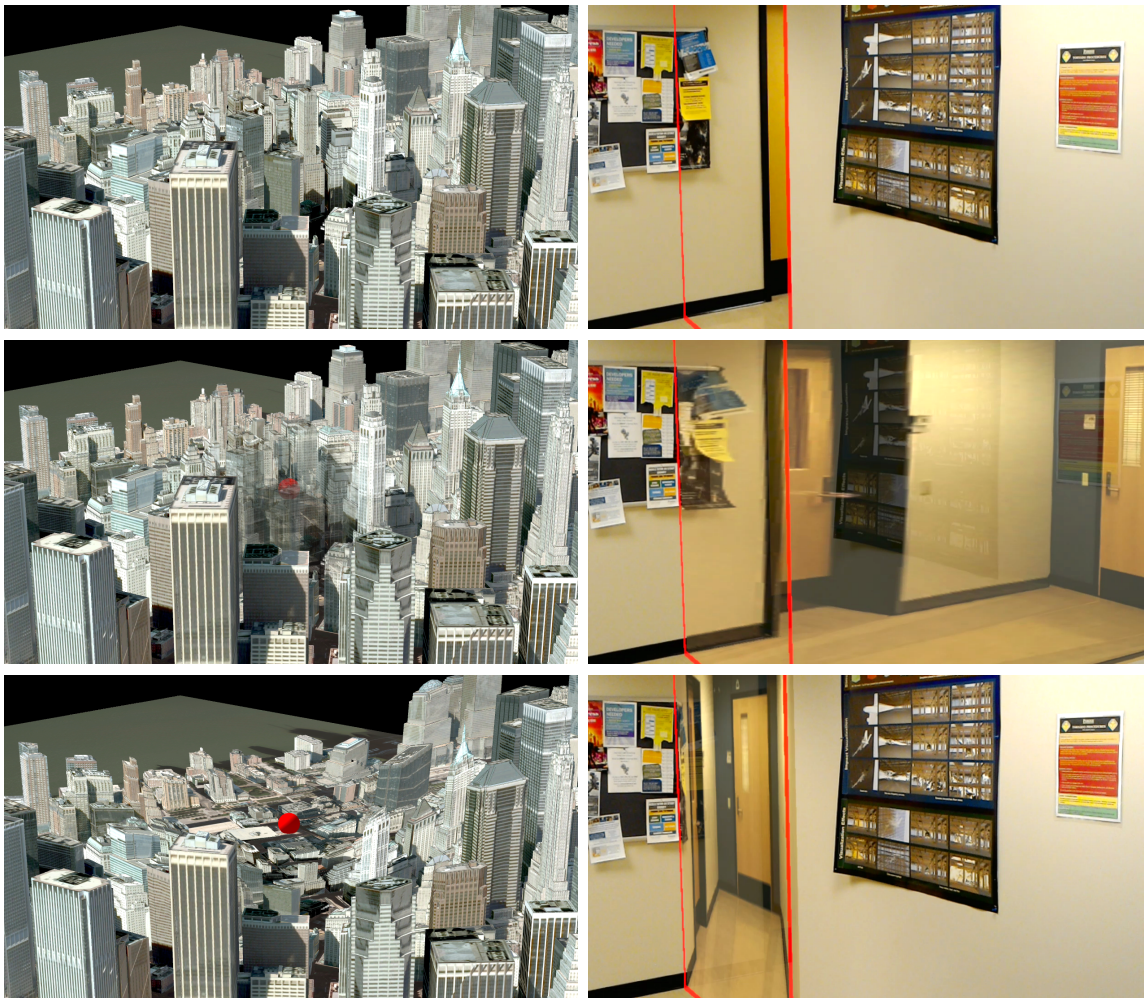


Fig. 3.9.: Comparison between conventional (top row), X-ray (middle row), and multiperspective (bottom row) VR (left) and AR (right) visualizations. The street-level target (left) and the side corridor (right) are occluded in conventional VR and AR visualizations. Unlike the multiperspective visualizations, the X-ray visualizations do not convey the street-level location of the target and the nearby geometry.

with other approaches to enhance navigation, such as occlusion management based on X-ray, cutaway, and explosion visualization. We foresee that some applications, scenes, and tasks will be better served by one approach over another. Furthermore, hybrid approaches should also be investigated.

Our approach to VR navigation has potential advantages in a multi-user environment. Unlike non-linear VR navigation techniques such as teleportation, redirected walking, or non-linear mapping, our method strictly preserves mapping between physical and virtual spaces at unit scale for the main, user perspective. This opens up possibility for multiple users to simultaneously collaborate in a shared VR scene. Consider the urban VR scene visualized by multiple users. The same target can be disoccluded simultaneously for all users with multiperspective visualizations tailored to each individual user. Each multiperspective visualization shows the target at the same undistorted location, which allows the multiple users to refer to the target consistently.

Our work introduces a paradigm where the user receives visual assistance from additional perspectives, without the disorienting effect of abandoning the main perspective. Our work does not advocate for supplanting all user physical locomotion through the use of multiperspective visualization. Instead, our contribution is a flexible framework where the user can choose how much to walk and how much to rely on the multiperspective visualization. For example, in target tracking, some users relied completely on the multiperspective visualization which all but eliminated the need to walk, and some other users only relied on the multiperspective visualization to simplify their walking trajectories, in order to succeed at keeping the target in sight without having to walk fast and with abrupt turns. Inspired by the significant reduction of physical locomotion for some users in the urban VR scene, future work could study multiperspective visualization in VR or AR applications where the user is seated or almost stationary. Such applications include navigation between multiple virtual displays in a VR desktop, assisted surgical operations with multiperspective

augmented annotations, or improved situation awareness during operation of machinery or vehicles.

In the indoor scene, the multiperspective visualization shows what is visible beyond the first turn, and the user still has to walk to be at most one turn away from the target. Future work could examine extending the multiperspective visualization to show what is visible beyond the second and third turns away from the current user position, which will undoubtedly come at the cost of increased visualization complexity. Future studies are needed that will leverage the flexibility of our multiperspective rendering framework to investigate the optimal trade off between multiperspective disocclusion power and visualization eloquence.

4. ANCHORED MULTIPERSPECTIVE VISUALIZATION FOR EFFICIENT VR NAVIGATION

4.1 Introduction

In VR applications, a HMD tracked with six degrees of freedom supports using real walking for natural navigation, where there is an identity mapping between the user's physical and virtual motion. The user selects the desired view intuitively, by walking to translate the viewpoint, and by rotating their head to change view direction. However, real walking navigation presents several challenges. One challenge is the fact that the real world space hosting the VR application is typically smaller and of a different shape compared to the virtual space, which can prevent the user from reaching some desired viewpoints. For example, a desired viewpoint might coincide with real-world furniture, it might be beyond the walls of the real world room, or it might be high up, on a higher level of a multistory virtual world that is hard to reach.

Another challenge is that in complex virtual world scenes occlusions limit how much the user can see from any given viewpoint. Comprehensive exploration requires translating the viewpoint to circumvent occluders and to gain line of sight to all potential ROIs. When a potential ROI turns out to be of no interest, the user has to retrace their path and to explore the next one. Such sequential scene exploration is inefficient. Furthermore, when scene understanding depends on seeing several ROIs simultaneously, or on visualizing dynamic, possibly evading targets, sequential scene exploration is ineffective.

Another reason why real walking might not always be desirable is based on ergonomics considerations. For some applications, the user might prefer not to expend the energy needed to navigate the VR world by always walking and rotating their head in the real world. In other words, for applications where the experience of actual

physical locomotion is not essential, users might prefer navigation interface constructs that allow them to see more with less physical effort in a shorter amount of time.

Many approaches have been investigated for overcoming these challenges of using real walking for navigation in VR. One promising approach is based on MPV, which relies on images that integrate samples captured from multiple viewpoints. Consider a virtual scene with two corridors intersecting at a right angle. Using a conventional visualization, a user has to translate the viewpoint up to the intersection to examine the side corridors in search of an ROI. If, on the other hand, an MPV shows not only the main corridor but also the side corridors, the user can examine the side corridors from their current location, which avoids the unnecessary navigation to the intersection when the side corridors turn out to be empty. Similarly, MPV can let the user see distant parts of the scene, without having to move beyond the walls of the real world space hosting the VR application. An MPV can also let the user examine two potential ROIs simultaneously, in parallel, even when no conventional visualization can show both ROIs at the same time.

Harvesting these potential advantages of MPV in the context of VR navigation requires solving two problems: (1) to design an MPV that is effective, i.e. that has the high information payload needed for navigation efficiency, but that remains easy to interpret by the user, and that does not induce user disorientation or motion sickness; (2) to devise navigation interface elements that allow the user to invoke their MPV superpower intuitively, in order to benefit from the additional perspective quickly and to the fullest extent.

In this chapter we present anchored multiperspective visualization, a novel multiperspective visualization method designed to improve VR navigation efficiency. Our method was designed based on the following principles: (1) the MPV image should be continuous and non-redundant; (2) the MPV should show the near part of the scene with a conventional first-person visualization controlled through natural motion, anchoring the user; and (3) the MPV effect should be controlled with user motions

reminiscent of natural motion, by tethering the secondary perspective selection to the user's head rotations and translations.

We have designed three types of anchored MPV. The first type allows the user to achieve a lateral disocclusion effect (Fig. 4.1, top) The user cannot see down the right side corridor with a conventional visualization (left). The MPV (right) integrates a secondary perspective into the main user's perspective, allowing the user to see down the right side corridor. The secondary perspective is controlled by the user translating their head to the left as if to look around a corner. The small head translation is amplified and applied to a secondary viewpoint that swings into place to reveal the side corridor. The user view change is used directly, without amplification, to render the nearby geometry, which remains in agreement with the user's proprioception to anchor the user.

The second type of anchored MPV allows the user to achieve a vertical disocclusion effect (Fig. 4.1, bottom). The user cannot see on top of the ledge in a conventional visualization (left). The MPV (right) integrates an additional perspective, with a high up viewpoint, to reveal the object on the ledge. The secondary perspective is controlled by the user by getting up on their tiptoes as if to examine a tall shelf above eye level. The small vertical user viewpoint translation is amplified and applied to a secondary viewpoint that translates up the necessary amount to see on top of the ledge.

The third type of anchored MPV allows the user to teleport from one location to another. MPV disoccludes parts of the scene not visible from the main user viewpoint, but it does not and should not produce a visualization that shows the entire scene. Consequently, the need to quickly move directly to a distant location of the scene remains even in MPV navigation. We have designed an anchored MPV teleportation method that proceeds in two stages, evocative of how a caterpillar moves (Fig. 4.2). First, the secondary viewpoint moves forward, getting closer to the far part of the scene, translating from the origin to the destination, while the primary viewpoint

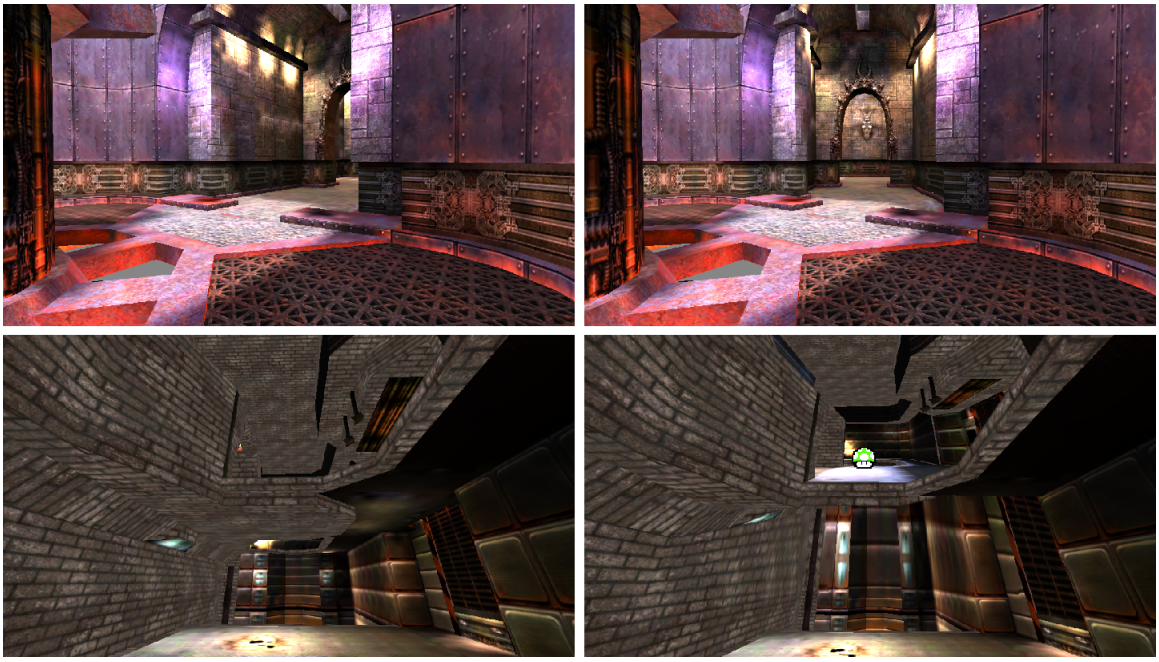


Fig. 4.1.: Top: lateral disocclusion effect. The side corridor is occluded in a conventional visualization (left), and visible in our anchored MPV (right). The disocclusion effect was deployed by the user with a small left translation of their head. The MPV shows the near part of the scene conventionally, anchoring the user. Bottom: vertical disocclusion effect. A conventional visualization does not show on top of the ledge (left), whereas our anchored MPV does (right). The disocclusion effect was deployed by the user with a small upward translation of their head achieved by getting up on their tiptoes. The MPV shows the ledge and the walls in front of the ledge conventionally, anchoring the user.

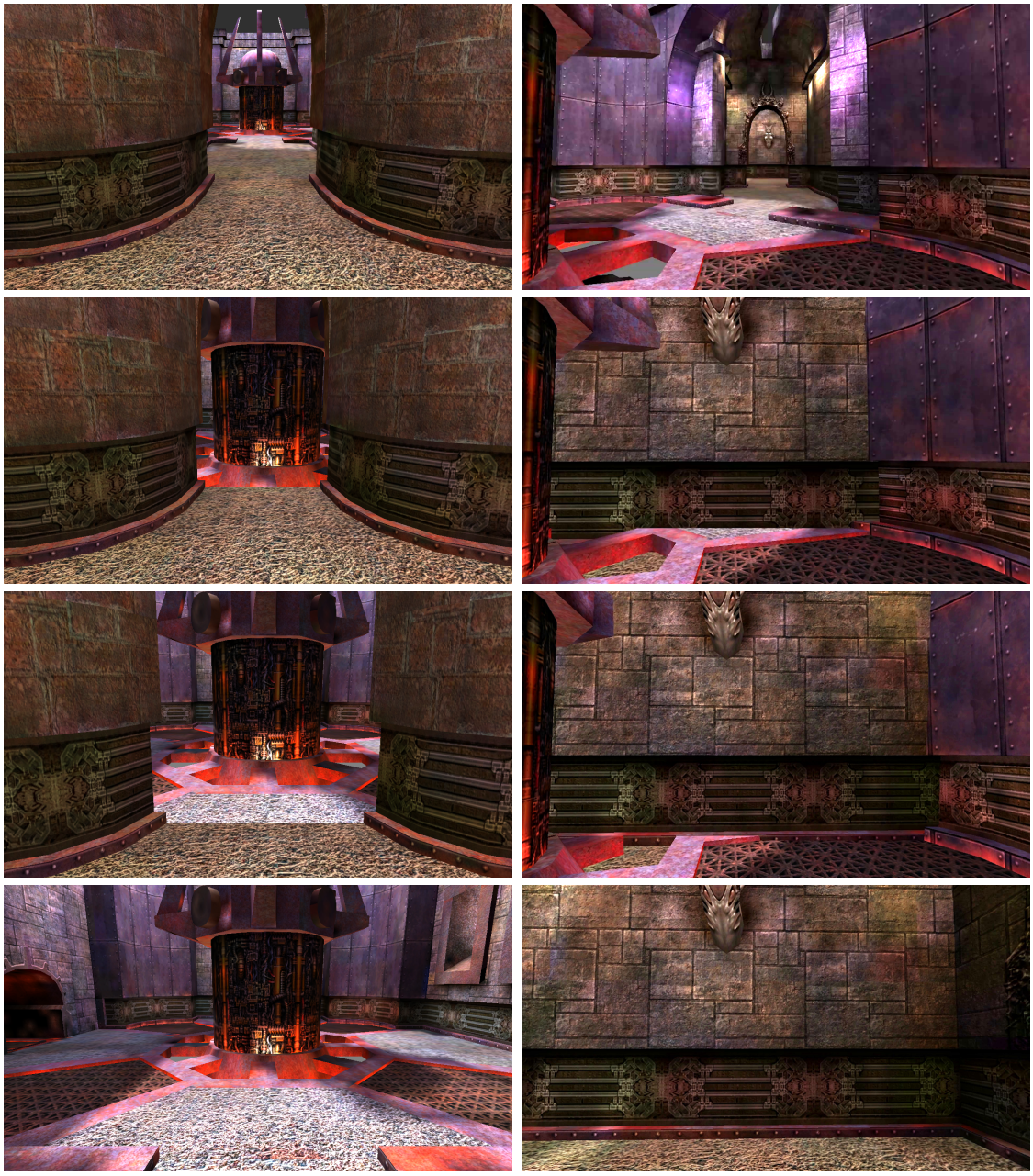


Fig. 4.2.: Two-stage MPV teleportation concept. In the first stage (top two images), the primary perspective stays locked on the origin, anchoring the user, while the secondary perspective translates to the destination. In the second half (bottom two images), the secondary perspective stays fixed, anchoring the user, while the primary perspective moves to assume the secondary perspective.

doesn't move, remaining at the origin. Second, the primary viewpoint moves forward to the secondary viewpoint, while the secondary viewpoint doesn't move.

We have conducted a user study to detect and quantify any VR navigation efficiency benefits brought by our anchored MPV method. 16 participants were divided evenly in a control group, who used conventional visualization, and an experiment group, who used our anchored MPV. Each participant performed a searching task and a matching task in each of two virtual environments: a single-story area of connected rooms, and a larger room with walkways suspended from the periphery walls, high above the room floor. For the first environment the experiment group participants had available our lateral disocclusion anchored MPV, and for the second environment they had available our vertical disocclusion anchored MPV. In all cases, all participants had the ability to teleport to any scene location to which they had line of sight. The experiment group used our MPV teleportation. The experiment group performed significantly better than the control group in the first virtual environment, achieving improvements in the metrics of distance traveled and number of teleportations. Cohen's d effect size of large and greater was observed with p-values below 0.05. In the more complex second virtual environment, the experiment group achieved improvements of medium Cohen's d effect size at p-values of 0.1 and less. Experiment group participants also reported improvements in spatial awareness and perceived navigation efficiency.

In summary, we make the following contributions: (1) a set of principles for designing VR navigation methods based on multiperspective visualization, (2) three anchored multiperspective visualization based on our design principles, one for lateral disocclusion, one for vertical disocclusion, and one for teleportation, and (3) a user study confirming the potential of our anchored MPV to improve VR navigation efficiency.

4.2 Prior Work

In VR, a preferred scene navigation modality is actual user locomotion in the physical space, which is translated to matching view changes in the virtual world.

However, the physical space typically differs considerably from the virtual space. Due to this mismatch, some virtual viewpoints become inaccessible. In Section 4.2.1, we discuss this challenge and prior work aimed at alleviating it. Another challenge arises from the reduction in visualization efficiency due to occlusions of ROIs by scene geometry, forcing the user to search for an unobstructed line of sight through extensive viewpoint navigation. In Section 4.2.2 we review prior work for improving VR navigation efficiency using the multiperspective occlusion management approach.

4.2.1 VR Navigation Challenges

The most intuitive VR navigation is an identity mapping between physical and virtual motion [42,43]. One common problem is that the physical space is considerably different than the virtual space. Usually the physical space is more restricted than the virtual world.

To fully explore the virtual world, the real and virtual locomotion must purposefully diverge to allow sufficient virtual motion while limiting physical motion. One approach is teleportation, which allows the user to designate a destination in the virtual world, and then to instantly relocate to that destination [69]. The visualization is discontinuous as the user changes location instantaneously, without any indication of the position of the destination relative to the origin. Therefore, the user needs some time to reorient themselves after arriving at the destination. A technique to reduce this discontinuity is to translate the user from the origin to the destination along a straight path [70]. However, a slow translation might cause nausea, as the user's viewpoint changes without any perceived acceleration, while a fast translation does not resolve the visualization discontinuity issue. In practice, a visual "blink", i.e. fade-out followed by fade-in, is applied as the viewpoint translates, to minimize nausea while providing some visual connection between the origin and the destination [71].

Another approach is artificial, or free, locomotion, where the user relies on input devices such as joysticks or keyboards to navigate the view beyond the tracked head

pose. The divergence between virtual and physical motion is thus directly controlled by user input. This method preserves visualization continuity, so spatial awareness is not compromised. However, due to the detachment of the user's virtual movement from their physical movement, the artificial locomotion method induces more motion sickness compared with teleportation-based methods [72]. Specifically, the visual and physical senses of acceleration are out of sync. One technique for alleviating nausea is to limit the user to discrete artificial locomotion steps, which are enacted abruptly to break the sensation of artificial acceleration. However, larger steps impact spatial awareness, while smaller steps incur frequent visual discontinuity [71].

Other approaches hide the mismatch between the physical and the virtual worlds by deviating from the tracking data, for example by making the user cover long straight lines in the virtual world by walking in circles in the physical world [44, 45], by resetting user pose [48], and by modifying input gain [47, 73]. Another approach is to distort the virtual world to pack it tightly in the limited confines of the physical world [49]. An approach that blends physical and artificial locomotion is the treadmill approach, or the smart platform approach, where the user actually walks, but without covering large distances in the physical world [40]. The shortcomings of the approach are confusing motion divergences, tethering the user, and reliance on expensive and bulky hardware.

4.2.2 Multiperspective Visualization in VR

MPV is a class of visualization techniques that integrate multiple perspectives into the main user perspective. MPV originated in the visual arts, e.g. Picasso's Cubism, and is applied to achieving comprehensive visualization, such as a ski trail map showing simultaneously trails not all visible from a single viewpoint.

Earlier research work focused on relaxing the single center of projection constraint, but the sampling rays remain linear [19,60,61]. More recent work introduced piecewise linear or even curved sampling rays that provide the flexibility needed to go around

occluders to reach distant ROIs [23,24,29]. While relaxation of the constraints opened up more degrees of freedom in the camera model used to render the visualization, the camera model generalization also created the need for automatic and interactive constructors that provide the application with the desired disocclusion effect [1].

In VR occlusion management, MPV is also found to be an effective technique [2], while conventional desktop occlusion management techniques such as transparency and explosion visualizations face various challenges. Transparency techniques introduce visual clutter that scale with scene complexity [52], whereas the MPV approach does not introduce additional geometry and does not violate pictorial depth cues. Explosion techniques disturb scene geometry [34], which impact the user's spatial awareness in VR, whereas the MPV approach does not disturb surface connectivity.

Portal-based visualization is a technique closely related to MPV. It composites additional views of the scene within the main view in a picture-in-picture fashion [74]. In VR applications, it supports teleportation navigation where the user teleports to destinations revealed through the portal [75]. However, the teleportation destination, as viewed through the portal, is beyond the vista space [76]. The user is therefore unable to trace the path of teleportation.

Our MPV method increases scene exploration efficiency by giving the user a preview of ROIs that are occluded in a conventional visualization. Compared with portal-based visualization, our MPV incorporates disoccluded ROIs into the vista space, avoiding disorientation due to untraceable teleportation [77]. Compared with the prior work in MPV navigation [2], our MPV supports both lateral and vertical disocclusion. It provides more versatile and at the same time more intuitive ways of controlling the additional perspectives, and it allows the user to assume seamlessly any additional perspective revealed by MPV using teleportation. The user's spatial awareness is further increased by visually anchoring the user in the scene as the additional perspectives are deployed and retracted, and during teleportation.

4.3 Anchored Multiperspective Visualization

In this section, we first discuss in more detail our three principles for the design of effective multiperspective visualization for VR navigation, and then we present our three methods for anchored MPV.

4.3.1 Design Principles for Effective Multiperspective Visualization in VR

(1) *MPV image continuity and non-redundancy*

This first principle encapsulates general concerns for achieving effective MPV, irrespective of the VR context. An MPV has to be continuous, i.e. points that are close in 3D should project to nearby image locations. This concern disqualifies MPVs obtained through a discontinuous collage of individual perspectives, or through parallel visualization with multiple disconnected rectangular images. An MPV also has to be non-redundant, i.e. it should not show a part of the scene multiple times. Continuity and non-redundancy are necessary conditions for obtaining an MPV that can be parsed by the user without the disadvantage of a significant cognitive load, which is particularly important in the VR navigation context.

(2) *Primary perspective MPV anchoring*

This second principle ensures that, as the conventional visualization morphs into an MPV, and as the MPV parameters are changed interactively, there is always a significant part of the image that is unaffected by the MPV effect, and that the unaffected part of the image corresponds to the space surrounding the user. The user's visual system relies on this primary visualization of nearby geometry, in sync with their own primary perspective, to remain in agreement with the motion perceived by the user, and to dissociate from the distant parts of the scene that move incongruently with the perceived motion, both of which contribute to preventing disorientation and motion sickness.

(3) *Natural secondary perspective navigation*

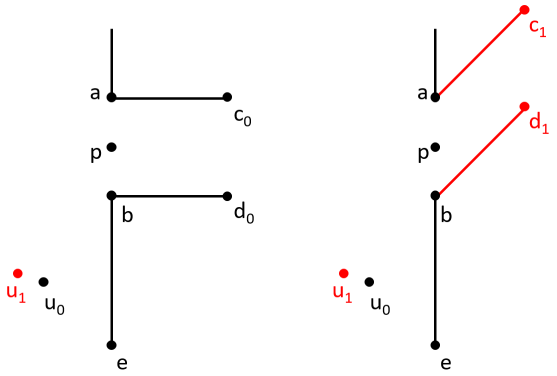


Fig. 4.3.: Lateral disocclusion through anchored MPV. A conventional visualization from viewpoint u_0 does not show the side corridor (left). A small left translation of the head from u_0 to u_1 deploys a disocclusion effect that shows inside the side corridor (right).

An MPV has a significantly higher number of degrees of freedom than a conventional visualization, with each additional perspective introducing six more extrinsic parameters. This third principle prohibits complex navigation interfaces that ask the user to manipulate a high number of degrees of freedom individually, and mandates allowing the user to control the secondary perspectives with natural motions similar to the ones used to control the main user perspective in conventional VR.

4.3.2 Anchored MPV for Lateral Disocclusion

A frequently needed disocclusion effect is to see around an occluder, e.g. to see around a tree, or to see through a side opening, e.g. through a window in a house facade. Such a lateral disocclusion effect can be provided by integrating a secondary perspective from a viewpoint that has line of sight around the tree or through the window. We provide a lateral disocclusion effect as follows.

Given a virtual scene, we define a set of vertical rectangles in the scene to serve as portals to guide the lateral disocclusion effect. The portals are defined where the user is likely to benefit from disocclusion, e.g. at the doorways that connect various sections of the scene, or between an occluder and nearby walls, such as a column in a

middle of a room. The portals are not rendered, but when a user exploring the scene with a VR HMD sees the geometry spanned by a portal, the geometry changes color to indicate the availability of a disocclusion effect. The user activates the disocclusion effect with a controller button. The activation itself does not change the visualization to an MPV. After activation, lateral translations of the user's head as recorded by the HMD will deploy a secondary perspective that sees through the portal.

In Fig. 4.3 the user's initial viewpoint is u_0, looking at portal ab. In the conventional visualization (left), the user cannot see deep inside the portal from u_0. Once the user activates the disocclusion effect of the portal, a subsequent left translation of the user viewpoint rotates the geometry behind the portal plane about the pivot point p, which is the center of the portal rectangle. The rotation gives the user line of sight perpendicularly through the portal, e.g. swinging the side corridor wall vertices c_0, d_0 to c_1, d_1, respectively (right). The rotation angle is proportional to the user's lateral head translation, and the gain is tuned such that a small amount of translation u_1u_0 (e.g. 20cm) is sufficient to see down the portal. The small translation wouldn't have been sufficient to see down the portal with a conventional visualization, i.e. the user would see only marginally more inside the portal from u_1 as compared to u_0 (left). The small translation is amplified by our lateral MPV disocclusion effect to introduce the necessary second perspective on the geometry beyond the portal plane.

The rotation angle is capped to the value needed to the see down the portal. Any geometry vertices or fragments that cross the portal plane when rotated are discarded (i.e. they are not drawn), which does not create artifacts as this geometry wouldn't have been visible anyway due to the side corridor walls. Vertex projection is continuous and non-redundant, which enforces the first design principle. Nearby geometry, i.e. the part of the eb wall seen by the user, is drawn conventionally, from the primary perspective, which anchors the user, enforcing the second principle. The additional perspective is deployed by the user translating their head to the left, and by slightly panning the view to the right in order to keep the portal in the center of the image, which is the natural motion the user would make if they were close to

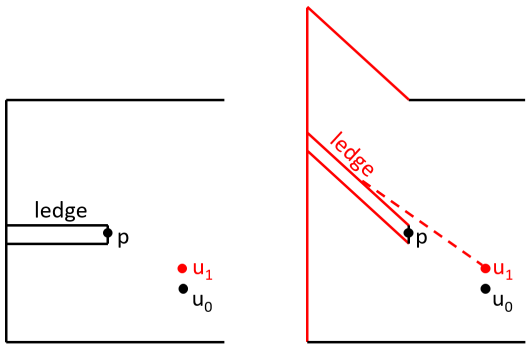


Fig. 4.4.: Vertical disocclusion through anchored MPV. A conventional visualization does not show on top of the ledge (left). The user tiptoes to generate a small upward head translation from u_0 to u_1 that deploys a disocclusion effect that shows on top of the ledge (right).

point b and wanted to look inside the portal, so the interface is in agreement with the third principle.

4.3.3 Anchored MPV for Vertical Disocclusion

In addition to lateral disocclusion, an explorer of a 3D virtual scene might also want to be able to see on top of horizontal surfaces that are suspended above the user's eye sight. Given a virtual scene, we define a set of ledge edges. Like in the case of portals, when a VR explorer has a predefined ledge edge into view, a highlight alerts them to the availability of a vertical disocclusion effect, and the user can activate the effect with a controller button.

Fig. 4.4 shows the conventional visualization of a scene with a ledge (left) and the same scene with our vertical disocclusion MPV effect (right). The initial user viewpoint u_0 is too low to see on the ledge. A small vertical translation of the user viewpoint to u_1 brings in an additional perspective that sees on the ledge. The disocclusion effect is implemented as a rotation of the geometry beyond and above the ledge, such that the new viewpoint u_1 is above the ledge plane, disoccluding the ledge. The viewpoint u_1 wouldn't have been high enough to disocclude the ledge in

a conventional visualization (left). The small translation u_0u_1 is amplified to achieve the vertical disocclusion effect.

Like in the case of lateral disocclusion, the vertex projection is continuous and non-redundant, and the user's view of the floor and of the ledge edge doesn't change, anchoring the user. Finally, the vertical up and down translation is controlled by the user's tracked HMD, who gets up on their tiptoes up to see atop the ledge, and back down to revert to a conventional visualization.

4.3.4 Anchored MPV for Teleportation

We allow the user to teleport between an origin and a destination viewpoint, as needed to change floors, rooms, and, in general, to overcome the constraints of the real world and of the tracking system. Teleportation is a rapid transition between the two viewpoints, which can induce user motion sickness because the user moves, i.e. "flies", without actually engaging in locomotion. Prior work suggests that the safest teleportation is a very abrupt one, but that is also the teleportation that disorients the user the most.

We have developed a teleportation that aims to alleviate these disadvantages. The user selects the destination with the cursor, which is always placed at the intersection between the view direction and the scene geometry. Therefore, the destination is selected with the HMD by changing view direction. The destination viewpoint is the point on the vertical through the cursor that is at the user's height above the ground. If this initial destination viewpoint is too close to a wall, the destination viewpoint is pulled back away from the wall to provide a meaningful view once teleportation is complete. The user triggers teleportation with a controller button.

Our anchored teleportation method is illustrated in Fig. 4.5. The origin and destination viewpoints are o and d. During the first phase (i.e. transition from left to middle in Fig. 4.5), the far perspective is brought closer, by translating the scene geometry beyond the cutting plane with vector od. The cutting plane is positioned

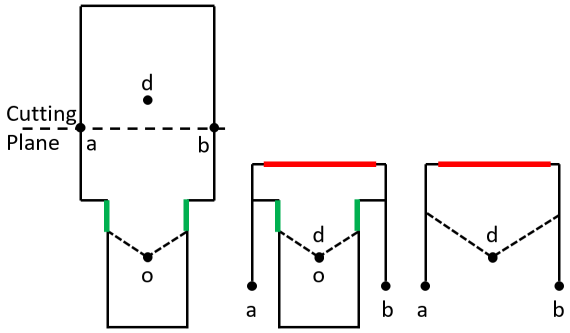


Fig. 4.5.: Anchored MPV teleportation. The viewpoint moves from o to d in two phases: first the part of the scene beyond the cutting plane is brought down, anchoring the user with the near part of the scene (green, from left to middle), and then the cutting plane is brought down, anchoring the user with the far part of the scene (red, from middle to right)

between the two viewpoints, splitting the distance between them at a fixed ratio. This first phase brings the far part of the scene closer, to be as close as it will be when the viewpoint is at d, but without pushing away the near part of the scene. The visualization of the near part of the scene (green) remains unchanged, which anchors the first phase of the teleportation. During the second phase, the cut plane is translated along the vector do, which makes the nearby geometry disappear from view. The visualization of the far part of the scene (red) remains unchanged in this second phase, which maintains uninterrupted anchoring.

Fig. 4.5 illustrates teleportation along a straight line, but the same procedure is followed if the user chooses to teleport using a lateral or vertical disocclusion MPV. The only difference is that once the second phase is complete, any residual distortion of geometry is gradually eliminated. Geometry is distorted only close to the portal or ledge planes, and typically the user desires to teleport deeply through the portal or beyond the ledge, so the geometry is undistorted off screen. Fig. 4.2, left, shows frames from a "straight line" teleportation. Fig. 4.2, right, illustrates teleportation into a side corridor, starting from the lateral disocclusion effect.

4.4 User Study

In order to evaluate the effect of our multiperspective visualization technique on VR navigation efficiency, we conducted a randomized user study, with the approval of our Institutional Review Board. Each participant performed three tasks in VR, and their actions were logged for subsequent analysis. After task completion, the users responded to a questionnaire that provided their subjective evaluation of task performance.

The participant wore a VR HMD (i.e. Windows Mixed Reality headset [78]). The VR HMD performs six degree of freedom SLAM-based tracking of the pose of the participant's head. In addition, the participant used a motion-tracked hand-held controller to enable interactions with the virtual world through actions such as

pointing and clicking buttons. The HMD displays stereoscopic image pairs rendered from viewpoints offset by the interpupillary distance to provide stereoscopic depth perception.

4.4.1 Participants

A total of 16 participants (14 male) completed our study. The participants were graduate students of ages between 23 and 37. They were randomly assigned to experiment and control groups of 8 participants each. Participants in the control group performed all tasks using only conventional, single-perspective visualization, while the participants in the experiment group performed all tasks using our anchored MPV.

The between-group design was chosen over the within-subject design to avoid any learning effect when repeating tasks from one condition to the next. Another reason is to mitigate the fatigue factor that can affect performance in extended task performance in VR environments–with the between-group design a participant's involvement time is reduced in half. However, the effect of prolonged usage of MPV navigation is an important avenue for future research, which is discussed further in Section 4.5.

4.4.2 Tasks and Evaluation

The participants performed tasks that required extensive virtual locomotion in two VR scenes using environments adapted from the Quake 3 Arena [79]. The first VR scene (Scene 1) is a single-story indoors area consisting of a set of rooms connected by corridors (Fig. 4.6, left). Two of the rooms contain cylindrical pillars in the center, which partially occlude the interior of the rooms regardless of the participant's current viewpoint. The second VR scene (Scene 2) is a large 3-story building where the center is an inaccessible tower, while the levels are connected by walkways attached to the perimeter walls (Fig. 4.6, right).

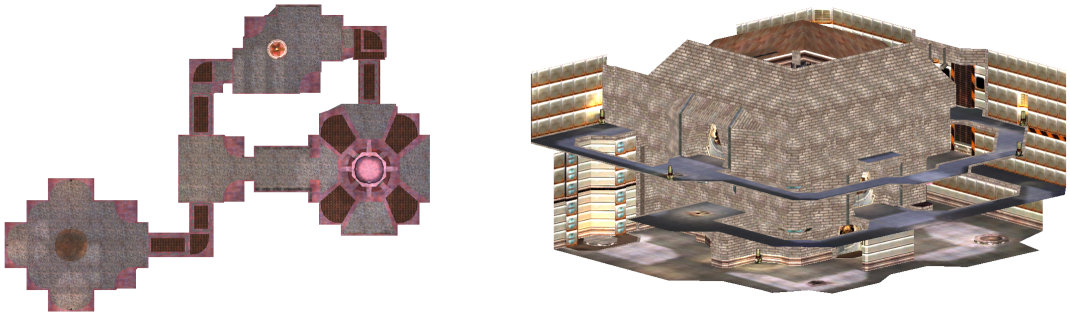


Fig. 4.6.: VR scenes used in the user study. Left: Scene 1 is a set of rooms connected by corridors. Right: Scene 2 is a 3-story building with multi-level walkways.

Each participant was required to perform 3 tasks: two search tasks, each in a different VR scene, and one pair matching task. Before beginning each task, the participant completed a short warm-up exercise which is similar to the actual task, but differs in content and is much shorter. This warm-up period ensured that the participant correctly operated the HMD and the hand-held controller, and that the test procedures were clear. Participants received no other training beyond this warm-up period.

The participant's performance was evaluated from recorded data using objective metrics, which were unknown to the participants. During the performance of each task, the participant's tracked physical HMD pose was logged, along with any interaction events such as initiating teleportation, acquiring a target object, or matching a pair of target objects. The logs were processed to extract metrics of interest. Additionally, each participant was asked to respond to a questionnaire after completion of the tasks. The questionnaire recorded subjective evaluation of performance and comfort.

4.4.3 User Interface for Locomotion and Anchored MPV

In both Scene 1 and Scene 2, the user was free to employ real walking by moving within a $2m$ by $2m$ physical space. To navigate beyond the limits of the physical space, the user employed teleportation. The user pointed the hand-held controller at the intended destination, and then clicked the controller button to initiate the teleportation. (Fig. 4.7, left). In Scene 2, the user could additionally access higher floor walkways by designating the walkway edges as a teleportation destination. As the user points the hand-held controller at a walkway edge, the edge is highlighted to indicate that the edge is a valid destination (Fig. 4.7, right). The user can also select any visible lower floor surface. As the controller button is clicked, the virtual viewpoint is teleported to the destination. The teleportation is not completed instantly, but the virtual viewpoint is translated to the destination at a fast $40m/s$ straight

Fig. 4.7.: The user selects the teleportation destination using a hand-held controller. Left: In both VR scenes, the user teleports by pointing a "laser" beam at the destination and clicking using the controller. Right: In the second VR scene, the walkway edges that are eligible as teleportation destinations are highlighted in blue when swiped with the laser beam.

line velocity. During the translation, participants in the control group experienced the conventional "blink" visual effect, whereas participants in the experiment group experienced the anchored MPV teleportation described in Section 4.3.4.

Our lateral disocclusion anchored MPV (Fig. 4.1, top) was accessible to experiment group participants performing tasks in Scene 1. The anchored MPV is activated automatically as the user gazes at virtual portals defined by corridor archways or connecting cylindrical pillars to side walls. The available archways and pillars in Scene 1 are predetermined prior to the study. Once the MPV is activated, the user moves laterally in small amounts to rotate the secondary perspective horizontally.

Our vertical disocclusion anchored MPV (Fig. 4.1, bottom) was accessible to experiment group participants performing tasks in Scene 2. The vertical disocclusion anchored MPV is manually activated. First, the user designates a walkway edge on a different floor by pointing with the hand-held controller and holding down the button. Then, with the button held, the user tiptoes or crouches a small amount to raise or lower the secondary viewpoint. Due to the need to hold down the button for activating the anchored MPV, the user of the MPV is required to double-click the controller button in order to initiate teleportation in Scene 2. Any new area revealed by the anchored MPV is also a valid destination for teleportation.

Search Tasks (Search 1 and Search 2) The search tasks Search 1 and Search 2 were performed in Scene 1 and Scene 2 respectively. In both search tasks, the participant was asked to find target objects in the form of gold coins placed in the VR scene. There were 6 coins that appeared one after another. Their locations were unknown to the participant, but were identical for all participants. A coin disappeared once it was found and collected by the participant by getting within $1m$ of the coin, either through real walking or through teleportation. An audio cue was triggered at the collection of a coin to notify the participant. The task was complete when all coins are collected.

Pair Matching Tasks (Match) The pair matching task (Match) was performed in Scene 1. In this task, the participant was asked to identify target objects in the form

of colorful mushrooms, of identical color pattern. There were 8 objects in 4 distinct color patterns, whose placement was identical for all participants. The participant first pointed to one visible target object and selected it by clicking a button. When the participant clicked on the second object of identical pattern, the pair was considered matched and it disappeared from the scene, with an audio cue, The task was complete when no targets remained.

Post-Performance Questionnaire Each participant was asked to respond to a questionnaire after completing the three tasks. The purpose of the questionnaire was to evaluate the participant's perception of their own performance and of of our MPV technique. The participant responded to each of three statements "You always felt present in the virtual world", "Your spatial awareness was maintained while moving around", and "You could reach any intended destination efficiently", by choosing an answer on a 1 to 5 scale, with 1 meaning "Not at all" and 5 points meaning "Very much".

4.4.4 Results and Discussion

We analyzed each participant's recorded logs to measure performance along several metrics. The most relevant metrics for this study are the metrics for navigation efficiency: 1) the number of times the participant initiated teleportation, 2) the accumulated teleportation distance, and 3) the time required to complete a task. These metrics were not revealed to the participants.

These metrics are evaluated for their effect sizes using Cohen's d [67], where qualifiers "small", "medium", and "large" are applied to cases where $d > 0.2$, $d > 0.5$, and $d > 0.8$, respectively. The qualifiers are extended to include "very small", "very large", and "huge" for $d < 0.01$, $d > 1.2$, and $d > 2.0$, respectively [68]. Statistical significance is evaluated using the two-sample t-test. We report the p-value to identify measurements that were due to chance.

Table 4.1.: Average number of teleportations per participant.

task	control	experiment	diff.	p	d	effect size
Search 1	83.4 ± 48.9	21.6 ± 18.5	61.8	< 0.01	1.7	very large
Match	51.8 ± 24.8	27.6 ± 22.7	24.1	0.03	1.0	large
Search 2	98.3 ± 71.5	50.3 ± 31.5	48.0	0.06	0.9	medium

Finally, we report the aggregate results of the questionnaire responses for discussion of subjective metrics which cannot be extracted from the task logs.

Number of Teleportations

Each teleportation incurs a discontinuity in the user's mental localization in the virtual world, after which the user must re-orient themselves. Therefore, it is desirable to minimize the number of teleportations. Furthermore, the ill-effects of teleportation can be reduced by improving visualization during teleportation.

We expect our anchored MPV to reduce the number of teleportations for two reasons. First, our MPV disoccludes ROIs and allows the user to plan their path more efficiently. Therefore, the user is able to avoid teleportating to destination only to find that it is not of interest. The second reason is that our anchored MPV extends the set of possible teleportation destinations to those areas newly disoccluded by the visualization. As the user is not limited to teleporting only to destinations with a line of sight, they are able to traverse at once a path that might require multiple teleportations using conventional visualization. Table 4.1 shows the result for the metric of number of teleportations. In Search 1 and Match tasks, which are both performed in the single-floor Scene 1, our anchored MPV significantly reduced the number of teleportations required to complete the tasks, with the p-value well below 0.05. The effect sizes are very large and large for tasks Search 1 and Match respectively. In the Search 2 task, which is performed in the more complex 3-story Scene 2, the results are positive with a medium effect, although with a higher p-value of 0.06.

Table 4.2.: Average distance traveled per participant, in meters.

task	control	experiment	diff.	p	d	effect size
Search 1	464 ± 135	222 ± 90	242	< 0.01	2.1	huge
Match	326 ± 103	228 ± 86	99	0.03	1.0	large
Search 2	677 ± 306	527 ± 236	150	0.1	0.5	medium

Table 4.3.: Task completion time, in seconds.

task	control	experiment	diff.	p	d	effect size
Search 1	113 ± 45	56 ± 29	57	< 0.01	1.5	very large
Match	89 ± 25	78 ± 25	11	0.2	0.4	small
Search 2	157 ± 71	175 ± 88	(18)	0.7	0.2	small

Accumulated Teleportation Distance

We analyze the accumulated teleportation distance because the majority of locomotion that is conducted in our VR scenes is through teleportation, therefore it is representative of the total amount of virtual viewpoint travel. We expect users of our anchored MPV to accumulate less traveled distance. This is due to the increased path planning efficiency as discussed in 4.4.4. Table 4.2 lists the average per participant distance traveled for each task. Significant improvements of huge and large Cohen d's effect sizes were observed for both Search 1 and Match tasks performed in Scene 1. This is in line with our expectation that the user is able to explore maps effectively with less required virtual locomotion. The result for task Search 2 performed in the more complex Scene 2 also shows an improvement of medium effect size. The statistical significance at $p = 0.1$ is not as strong as with tasks performed in Scene 1. However, the positive effect size suggests a more significant result is possible with more user study participants.

Task Completion Time

Table 4.3 reports the time our participants took to complete the three tasks. The experiment group had a statistically significant advantage for the first task. For the

second task, the experiment group was faster, but the advantage was not statistically significant. For the third task, the average completion time for the experiment group was longer than for the control group. From our observation of the participants during the experiment, we explain this based on a longer time the participant needed to engage the MPV interface. As shown in Table 4.1, the number of jumps is significantly lower for the experiment group even for the third task, indicating that the MPV is effective, except that using it takes longer. A direction of immediate future work is to improve the time effectiveness in which the vertical disocclusion MPV is used, by suggesting to the user the availability of the effect in more salient way during training (the blue highlight is sometimes easy to miss), and then by suggesting the tiptoeing mechanism that actually implements the MPV.

Questionnaire Responses

The final metric we used to compare the two participant groups was a compilation of the self-evaluation questionnaire responses (Table 4.4). The experiment group self-reported higher spatial awareness and higher navigation efficiency, while they reported a lower sense of actual presence in the virtual environment. It comes as no surprise that the experiment group had better spatial awareness than the control group as the MPV essentially provides a preview of the scene, without as much disorienting backtracking as required by the sequential exploration with a conventional visualization. Furthermore, our MPV was designed to anchor the user at all times, so the additional information presented did not come at the cost of confusing the user. Similarly, the improvement of navigation efficiency is a reasonable hypothesis based on the same arguments. We explain the decreased sense of presence by the fact that, like any MPV, our method upgrades the user from an uninterrupted immersive first-person view of the scene to an occasional second or even third-person monitoring of the scene. The user shifts fluidly viewpoint and their association with a specific

Table 4.4.: Overall subjective evaluation.

	control	experiment	diff.	d	effect size
Spatial awareness	3.3 ± 1.3	4.3 ± 1.0	1.0	0.7	medium
Efficiency	4.1 ± 1.5	4.6 ± 0.5	0.5	0.4	small
Presence	4.0 ± 1.1	3.5 ± 1.1	(0.5)	0.4	small

location in the scene, and the consequent decrease in sense of presence is a reasonable trade-off towards gaining navigation efficiency.

4.5 Conclusions and Future Work

We have presented a novel method for multiperspective visualization for Virtual Reality that promises to improve navigation efficiency. Our method visualizes the scene with images that show more than what is visible from a single viewpoint, by integrating additional perspective continuously and non-redundantly. Another goal of our MPV is to always anchor the user by showing part of the scene geometry conventionally, from the user's first-person view. Finally, MPV navigation should remain as intuitive as possible, by allowing the user to control the additional perspectives through the tracked HMD. We describe anchored MPV techniques for lateral disocclusion, for vertical disocclusion, and for teleportation. The MPV benefits have been confirmed in a user study.

Additional user studies should be conducted to explore in depth the subjective effects of MPV navigation. Our post-performance questionnaire provided preliminary insights to the users' perceptions, but the study is not tailored to measure subjective effects such as visual quality, cognitive effort, and user comfort. Especially, tasks which require prolonged usage of MPV navigation should be designed to examine any cumulative effect of simulator sickness, even though no test participant expressed discomfort at any point during the study. At the same time, these longer tasks allow examination of any training effects as participants become familiar with the user interface.

One direction of future work is to find automatically the places in the scene where disocclusion effects are useful. This is in view of the limitation that the virtual portals needed to be manually marked in Scene 1 of our user study. One option is to preprocess the scene, and another option is to decide on the potential for disocclusion on the fly, based on the current frame.

Another direction of future work is to investigate the anchored MPV benefits in the context of dynamic, and even evading targets, which place even more stringent requirements on the quality of the disocclusion effect and on the intuitiveness of the interface for deploying it. These requirements are particularly relevant to gaming applications, where targets could follow complex strategies, or they could be other humans. Furthermore, there is interplay between leveraging visualization to facilitate navigation, and designing visualization for game mechanics. It is worth studying how to optimize for both sets of goals in the design space for VR visualization. As VR interfaces strive to become mainstream and to move beyond entertainment and into day to day use, a scenario that requires special attention is the sit at desk scenario, for which multiperspective visualization might be particularly well suited.

5. RGBD TEMPORAL RESAMPLING FOR REAL-TIME OCCLUSION REMOVAL

Video streams are a widespread approach for visualizing real-world scenes in real time. However, video streams suffer from occlusion. In a dynamic scene, an object of interest, or target, can become temporarily hidden by other objects. In order to achieve an uninterrupted visualization of the target, one approach is to acquire the scene with multiple video streams, in the hope that the target is visible in at least one stream at any given time. Such a visualization with multiple streams implies a significant cognitive load for the user, who cannot monitor all streams in parallel, but has to examine the streams one at the time. Moreover, there is no guarantee that the target is visible in any one of the available streams.

Another approach is to rely on earlier frames to fill in the parts of the target that are occluded in the current frame. This inpainting approach for removing occlusions from video has the advantage of good visualization continuity, since the viewpoint does not change, and of simple acquisition, requiring only one video stream. Whereas there are real time video inpainting methods for removing occlusions of a static or of a rigid body target, prior art methods for deformable targets are too slow for real time performance.

In this chapter we describe a novel method for video inpainting to remove occlusions of a deformable target in real time. Our method takes advantage of the fact that, in many cases of interest, such as when the target is a walking human, the target deformation is coherent, and the range of deformations is small. Our pipeline segments the partially occluded target from the current frame and inpaints the missing pixels by resampling from a matching disoccluded target from an earlier frame. Our pipeline takes as input a single video stream enhanced with a depth channel, i.e. an RGBD stream, acquired by a stationary computer tablet with a structured light

(a) Video sequence A

(b) Video sequence B

Fig. 5.1.: Pairs of frames that illustrate our occlusion removal method: the original frame is shown to the left, and the frame output by our method is shown to the right. Our method runs at an interactive frame rate of 30 frames per second.

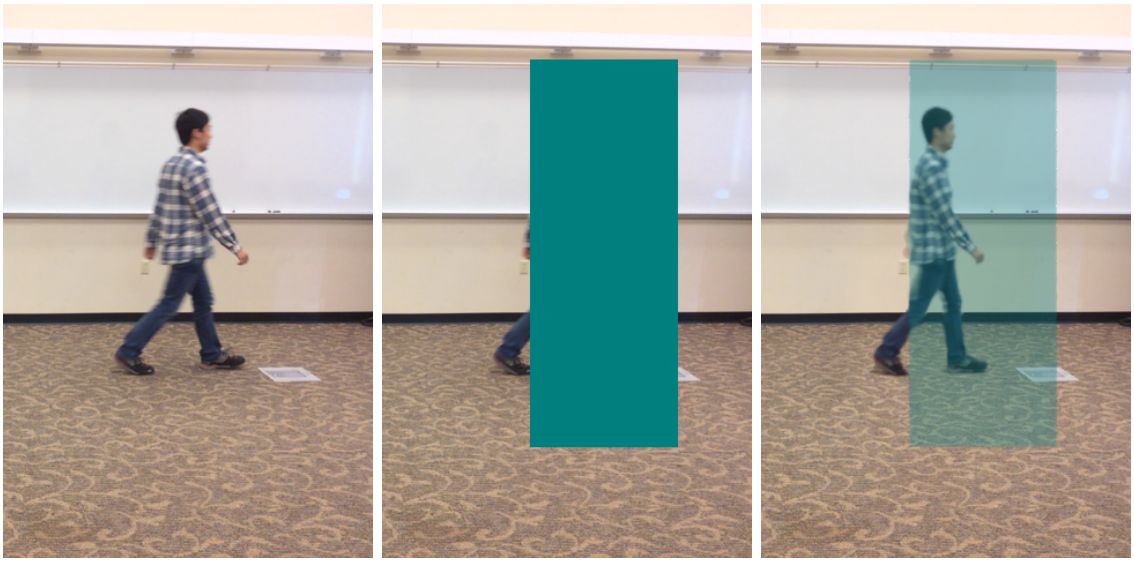


Fig. 5.2.: Video sequence C — Comparison to ground truth: original frame (left), synthetic occluder added to original frame (middle), and frame without occlusion generated by our method (right).

depth camera accessory. The depth channel enables faster and more robust target segmentation and matching.

We demonstrate our method in the case of a walking human who is occluded by other stationary or walking humans (Fig. 5.1). Fig. 5.2 compares our output to truth using a synthetic occluder. Our method does not rely on a known deformable model of the targets and occluders, and therefore it supports general target and occluder shapes. For example, in Fig. 5.1 lower, the target is a human pulling a wheeled bag. Our pipeline runs at interactive rates. We also refer the reader to the accompanying video that illustrates our method.

5.1 Prior Work

The removal of image defects, and the subsequent completion of the image by inpainting, is a well-known problem since the invention of photography [80,81]. Image defects such as scratches, dusts, text and scene clutter are cut out, and the remaining hole regions are inpainted using information taken from regions outside the hole, called

source regions [82]. Two fundamental approaches are low-level texture synthesis and and higher-level structure propagation.

Texture synthesis is concerned with computing a larger texture that is similar to a smaller initial sample texture. When large image defects such as unwanted objects are cut out, inpainting the large holes requires more than filling in with hole boundary colors. In these cases, a plausible background texture can be synthesized that is similar to surrounding source region texture, while maintaining continuity across the hole region boundaries [83,84]. The plausibility of the inpainted image depends on the background being self-similar. Performance is improved by patch-based inpainting, which uses source region patches instead of pixels [85, 86].

However, if cutting out the foreground reveals a background with salient geometry or non-repeating patterns, then the background structure must be extended into the hole region. The structure propagation approach extrapolates source region edges such as isophotes [87] or follows user provided guidance [85, 88] in order to inpaint plausibly. The structure propagation method is suited to inpainting cartoon-like image holes. Multi-level approaches that combine both texture synthesis and structure propagation are an active research area [89–91].

A video is a linear sequence of images. To remove an object from a video segment, the object has to be cut out from multiple frames where it is visible, leaving behind a spatio-temporal volume defining the hole region to be inpainted. Video inpainting is more challenging than image inpainting because a) a large number of images have to be inpainted; b) the source region where to search for the missing colors is larger due to the addition of the time dimension; c) the inpainted holes must maintain temporal continuity across frames, in addition to maintaining spatial continuity across hole boundaries [92–94].

Efficient video inpainting is possible, even in the case of dynamic camera poses, by propagating the mapping of inpainting pixels forward through successive video frames [95, 96]. However, the background in the hole region has to be static and with simple geometry, such as a few planes. In the case of static foreground and background

with significant depth separation, dynamic camera poses are leveraged to expose every part of the background through parallax. The static background can then be fully reconstructed and inpainted in every video frame [97]. Scene-space video inpainting removes the constraint on background simplicity by reconstructing complex scene geometry from depth images [98]. Although capable of filling in hole regions with complex background, the inpainting background scene points still need to be visible at other points in time at the same scene-space position, so the static background restriction is not removed. Thus, both of these approaches are ill-suited for occlusion removal when the occluded background is typically dynamic and deformable.

For the more general problem of inpainting dynamic and deformable targets, such as walking humans, it is necessary to find the best source region, and to map the current hole region to the source region, for every video frame. The global optimization approach takes the entire spatio-temporal volume of the video sequence as search space for source regions [86, 92, 99, 100]. By optimizing a visual quality energy function, the best mapping between the hole and source regions is obtained for inpainting the missing pixels. Although high visual quality is achieved, the performance is insufficient for interactive applications due to the large search space for source regions and the high cost to evaluate the energy function.

Object based methods improve performance by reducing the search space for source regions [101–104]. The target is segmented and tracked in the video. As occlusions occur, the hole region is inpainted with samples found in the target segment in video frames where the target is visible. Compared with the global optimization approach, the performance is improved when the target is small. However, to the best of our knowledge, video inpainting of deformable targets at interactive rates without user input has not yet been achieved.

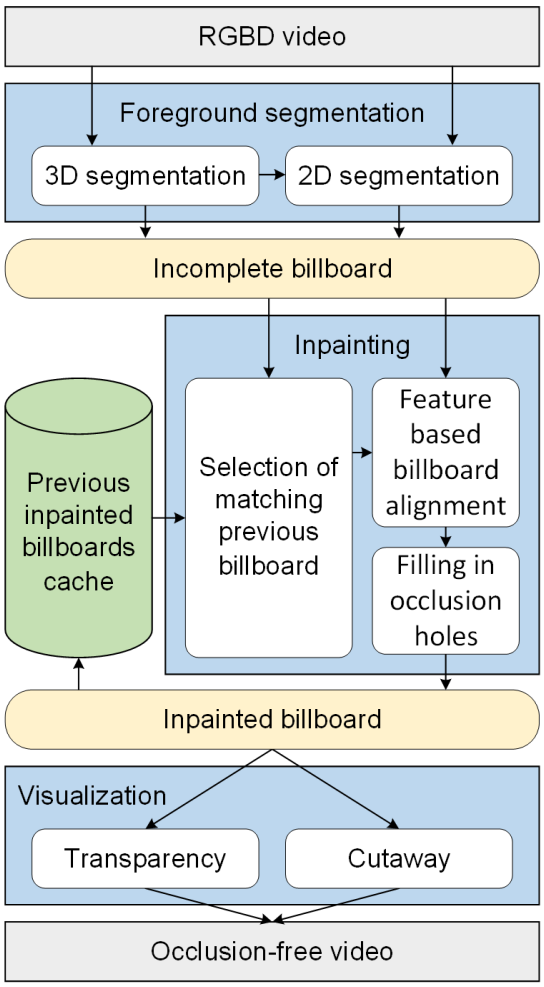


Fig. 5.3.: System Overview

5.2 System Overview

Our system pipeline is shown in Fig. 5.3. The system takes as input a single RGBD video stream acquired from a fixed viewpoint showing a moving and deforming target that becomes occluded. The system outputs a video stream in which the target is disoccluded. The system processes each frame with a pipeline that has three main stages: segmentation, which identifies the occluded parts of the target, inpainting, which fills in the occluded parts of the target, and visualization, which shows the frame with the target unoccluded.

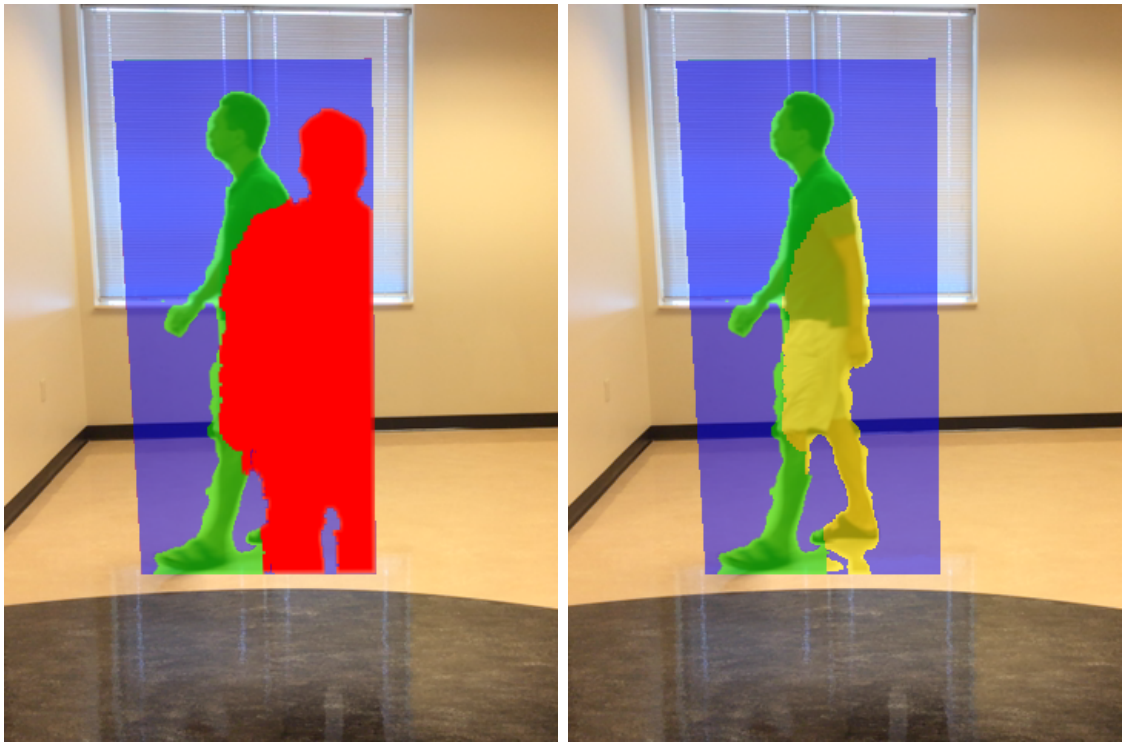


Fig. 5.4.: Illustration of incomplete (left) and inpainted (right) billboard for the case shown in Fig. 5.1. The parts of the target visible in the current frame are highlighted in green, the parts of the billboard that are neither occluded nor part of the target are shown in blue, the parts of the billboard that are occluded are highlighted in red, and the parts of the billboard that are inpainted are highlighted in yellow.

The frame is segmented both in 3D, by unprojecting pixels to a point cloud using the depth channel, and in 2D, using the color and depth channels. The output of the segmentation stage is an "incomplete billboard", which is a rectangular impostor in 3D that is texture mapped with the parts of the target visible in the current frame. The billboard texture has both color and depth channels. The incomplete billboard for the left pair in Fig. 5.1 is shown in Fig. 5.4. The segmentation stage is described in Section 5.3.

The incomplete billboard is inpainted using color and depth data from an earlier frame that shows the target in a similar pose. A matching earlier impostor billboard is found among the previously inpainted billboards, the current billboard is mapped to the earlier billboard using feature correspondences, and the occluded parts of the

current billboard are filled in by resampling from the earlier billboard. The inpainting stage is described in Section 5.4.

The inpainted impostor billboard is used for an unoccluded visualization of the target. The occluder can be shown semi-transparently (Fig. 5.1, upper), or it can be cut away (Fig. 5.1, lower). The visualization stage is described in Section 5.5.

5.3 Segmentation

The first step of occlusion removal is to identify the parts of the target that are occluded, which is done by segmenting the current frame (Alg. 1).

The algorithm for segmenting the current frame F_i takes as input a reference frame F_0 captured from the same view V_0 before the moving objects appear, and cylinders C_{i-1} bounding the moving objects that were computed at the previous frame. The algorithm proceeds in three major steps: 2D segmentation, 3D segmentation, and billboard construction. The 2D segmentation step (lines 3-5) partitions F_i into foreground and background by detecting changes in the depth channel with respect to F_0 (Fig. 5.5, top row).

The 3D segmentation step (lines 6-21) first computes a top view image I_{top} of the scene by unprojecting the foreground pixels to 3D points using the depth channel (line 9), and by splatting the 3D points into I_{top} using a camera with a downward view direction (line 10), see bottom left image in Fig. 5.5. Then, if F_i is the frame where the target first appears, the I_{top} is segmented into blobs using OpenCV's Simple Blob Detector algorithm [105], (line 12), and a vertical bounding cylinder is fitted to each blob (line 14). If the target was visible before (line 15), the blobs are tracked from frame to frame, instead of detected by searching blindly the entire top view image I_{top}. Each tracked blob BL_{ij} is found in I_{top} in the region where its bounding cylinder was found at the previous frame (line 17). If BL_{ij} is not present in I_{top} (line 18), the corresponding object is totally occluded and the its bounding cylinder C_{ij} is updated by extrapolating its position from the previous frames (line 21). We use

Algorithm 1 Segmentation

Input: current RGBD frame F_i, reference RGBD frame F_0, bounding cylinders C_{i-1} of moving objects in previous frame, camera view V_0.
Output: billboards BB_i of current frame, bounding cylinders C_i of moving objects in current frame.
// Step 1: 2D segmentation
for each pixel (u, v) in F_i **do**
  $FG(u, v) = F_i(u, v).z < F_0(u, v).z$
// Step 2: 3D segmentation
for each pixel (u, v) in FG **do**
  if $FG(u, v)$ **then**
    $p = \text{Unproject}(u, v, F_i(u, v).z, V_0)$
    $\text{Splat}(p, I_{top})$
if target appears for the first time **then**
  $BL = \text{DetectBlobs}(I_{top})$
  for all blobs BL_j in BL **do**
    $C_{ij} = \text{ComputeBoundingCylinder}(BL_j)$
else
  for all cylinders $C_{i-1,j}$ in C_{i-1} **do**
    $BL_j = \text{ComputeBlob}(I_{top}, C_{i-1,j})$
    if $\text{Size}(BL_j) > \epsilon_B$ **then**
      $C_{ij} = \text{ComputeBoundingCylinder}(BL_j)$
    else
      $C_{ij} = \text{ExtrapolatePosition}(C_{i-1,j})$
// Step 3: Billboard construction
for all cylinders C_{ij} in C_i **do**
  $BB_j.rectangle = \text{VerticalCrossSection}(C_{ij})$
  $BB_j.texture = Project(F_i, V_0, BB_j.rectangle)$
  for each pixel (u, v) on $BB_j.texture$ **do**
    if $F_i(u, v) == F_0(u, v)$ **and** $!FG(u, v)$ **then**
      $BB_j.texture(u, v).label = \text{background}$
    else
      $p = \text{Unproject}(u, v, Fi(u, v).z, V_0)$
      if p is inside C_{ij} **then**
        $BB_j.texture(u, v).label = \text{visible}$
      else if p is behind C_{ij} **then**
        $BB_j.texture(u, v).label = \text{background}$
      else if p is in front of C_{ij} **then**
        $BB_j.texture(u, v).label = \text{occluded}$

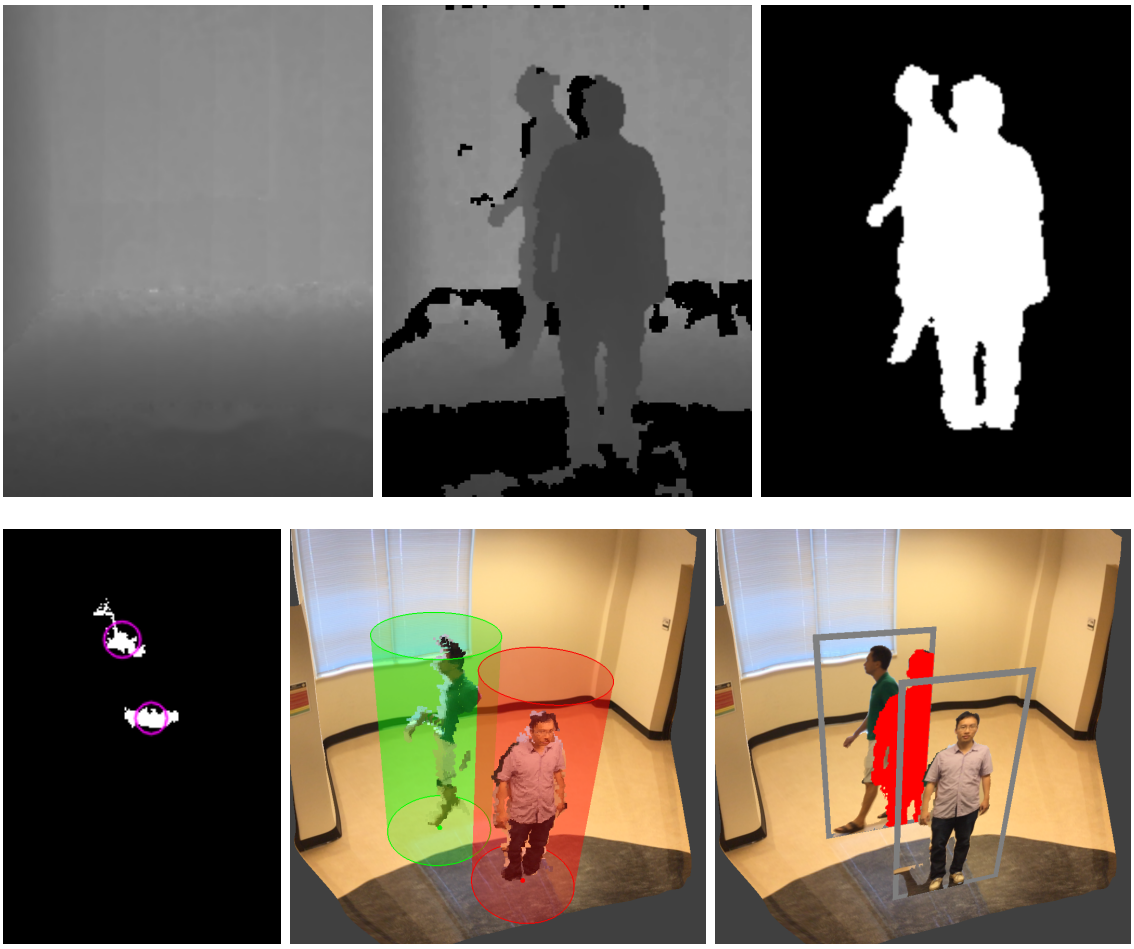


Fig. 5.5.: Illustration of the steps of the segmentation stage. Top: depth channel of reference frame F_0 (left) and of current frame F_i (middle), and output of 2D segmentation (right). Bottom: top view image of foreground objects used for 3D segmentation (left), and visualization from a translated viewpoint of bounding cylinders (middle) and of billboards (right).

a threshold ϵ_B below which the object is considered totally occluded. Furthermore, when the size of BL_J approaches the ϵ_B, the bounding cylinder is set by interpolating between the computed (line 19) and the extrapolated (line 21) cylinder (omitted for conciseness). The bounding cylinders are illustrated from a translated viewpoint in the middle image on the bottom row in Fig. 5.5.

The third step constructs an impostor billboard for each moving object based on its bounding cylinder. A billboard is a vertical rectangle defined by the cross section of the cylinder facing the camera (line 24), texture mapped with an RGBD image. The texture is computed by projectively texture mapping the current frame onto the rectangle (line 25), and then by labeling every texel as a visible part of the object (line 32, green in Fig. 5.4), as a visible part of the background (lines 28, 34, blue in Fig. 5.4), or as an occluded part of the object (line 36, red in Fig. 5.4). To label a texel, it is first compared to the reference frame F_0 (line 27). If the texel is distinct from the reference frame in the color channels or the depth value, then the labeling is decided based on the 3D segmentation of each object given by the bounding cylinder (lines 30-36).

5.4 Inpainting

In the inpainting stage, the occluded pixels of the target billboard are further classified as background pixels, which are set to transparent, and as occluded target pixels, which are inpainted using samples from a previous target billboard, as described in Alg. 2.

The algorithm takes as input the target billboard BB_t computed by the segmentation stage at the current frame i, the cache of previously inpainted target billboards $IBBC$, and the index b_{i-1} of the frame where the matching billboard was found at the previous frame. The algorithm outputs the inpainted target billboard IBB_i and the index b_i where the matching billboard was found.

Algorithm 2 Inpainting

Input: target billboard BB_t of current frame i, cache of previous inpainted target billboards $IBBC$, frame index b_{i-1} of matching billboard from previous frame.
Output: inpainted target billboard IBB_i and frame index b_i of matching billboard at current frame.
// Step 1: Selection of matching billboard
$\epsilon_{min} = \infty$
for $j = \lfloor b_{i-1} - w/2 \rfloor$ **to** $\lfloor b_{i-1} + w/2 \rfloor$ **do**
    $\epsilon = \text{MatchingError}(BB_t, IBB_j)$
    if $\epsilon < \epsilon_{min}$ **then**
        $\epsilon_{min} = \epsilon$
        $b_i = j$
// Step 2: Feature based billboard alignment
$BB_t.features = \text{FeatureExtraction}(BB_t.texture)$
$\text{FeatureMatching}(BB_t.features, IBB_j.features)$
Divide BB_t into square grid G
for each feature f in BB_t **do**
    c = cell of G containing f
    $d = f - f.match$
    if $||d|| < ||c.d||$ **then**
        $c.f = f$
        $c.d = d$
// Step 3: Filling in occlusion holes
for each missing pixel (u, v) in BB_t **do**
    $d = \sum_{c \in G} c.d \ \mathrm{K}(r, \sigma) / \sum_{c \in G} \mathrm{K}(r, \sigma)$,
    where $r = ||(u, v) - (c.f.u, c.f.v)||$
    $(u_j, v_j) = (u, v) + (d.u, d.v)$
    $BB_t(u, v) = IBB_j.(u_j, v_j)$
    $BB_t(u, v).label = IBB_j(u_j, v_j).label$
// Step 4: Update cache of inpainted billboards
$IBB_i = BB_t$
$IBB_i.features = \text{FeatureExtraction}(IBB_i.texture)$
$IBBC.Insert(IBB_i)$

The inpainting algorithm proceeds in three major steps: selection of matching billboard, feature based billboard alignment, and fill-in of occlusion holes.

To select the matching billboard (lines 3-9), the algorithm searches for the previous billboard that is most similar to the current billboard. A window of w billboards are considered, centered at the previously found best match. The width of the window w is empirically set at 0.3 times the distance, in number of frames, from the current billboard to the previously found best match. The index of the frame where the best matching billboard was found is recorded to center the search at the next frame (line 9). Given two billboards BB_t and IBB_j, the matching error is computed by minimizing the depth alignment error between the billboards as IBB_j is translated left-right and forward-backward with respect to BB_t. The depth alignment error for a left-right translation Δu and a forward-backward translation Δz is defined as the mean squared error between the depths z_t and z_j of the two billboards BB_t and IBB_j over the N overlapping pixels, as shown in Eq. 5.1.

$$\epsilon(\Delta u, \Delta z) = \sum_{u,v} (z_t(u,v) - z_j(u + \Delta u, v) + \Delta z)^2 / N \tag{5.1}$$

Fig. 5.6 gives a heat map visualization of the matching error for the 140 frame sequence used in Fig. 5.1, upper. A point (i, j) on the heat map corresponds to the matching error between a frame i and an earlier frame j. The black lines show the search window used by the algorithm, and the purple dots shows the best matches found for each frame. The black lines show that the search starts in the global minimum error valley. The plot shows that the period of motion is about 40 frames. The purple dots form a relatively straight line because the period does not change much, however, it is not constant. Our algorithm relies on the availability of a similar billboard, but it does not require that the motion be periodic.

If the target billboard computed by segmentation does not contain any occluded pixels, the inpainting algorithm stops. Step 1 has to be executed even for target

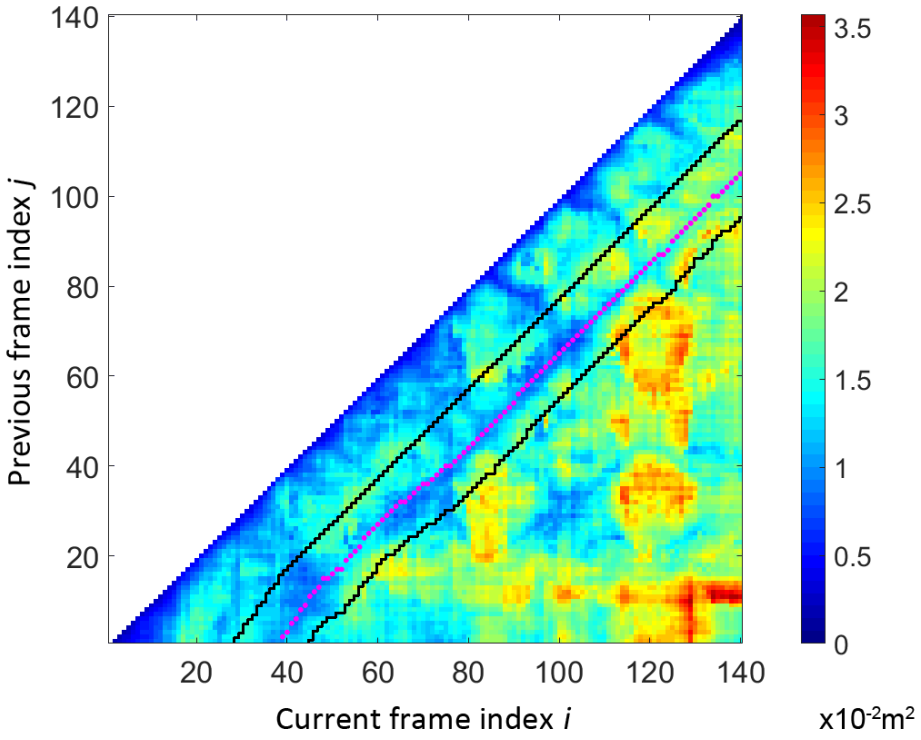


Fig. 5.6.: The heat map visualization of the matching error. The x axis is the current frame index i, and the y axis is the previous frame index $j, j < i$. The search window is the region between the black lines, and the best matches found for each frame are marked by purple dots. The entire heat map is shown here to cover all possible pairs of (i, j), whereas only the (i, j) pairs inside the search window are calculated in Step 1 of Alg. 2.

billboards without occluded pixels in order to update the frame index b_i where to start the next matching billboard search.

Before actual inpainting can begin, a mapping between the current and previous billboard has to be computed. The rigid alignment computed at Step 1 during the selection of the matching billboard is not sufficient for quality inpainting. At Step 2, a non-rigid alignment is computed based on color features (lines 10-19). ORB features [106] are extracted from the color texture of the billboard (line 11), and they are matched to the previously extracted features of the cached billboard IBB_j using the Hamming distance (line 12). The mapping between BB_t and IBB_j is computed efficiently with the help of a 2D grid G. The features of BB_t are assigned to cells in G (lines 14-19), which has a dimension of 20×40 cells, width×height. The grid G controls feature density in cells with great texture variation by culling all but one feature that are assigned to each cell. When multiple features are assigned to a cell, only the feature that is closest to its matched feature is kept (line 16-19).

Fig. 5.7 shows the billboard inpainted with and without non-rigid alignment. The left image shows extracted features in both the current and previous frames in colored circles. The matched features which survive culling are highlighted in cyan, while those that are discarded are colored red. The density of feature matches is effectively controlled by the culling process around the face and lower leg regions. The middle and right images show the inpainted billboards where the inpainted pixels are tinted red. With non-rigid alignment applied (right), the pixels around the shoulder are better aligned, while the pixels around the foot are unaffected and remain well aligned.

At Step 3, the grid with assigned features is used to compute the location from where to get color to fill in occlusion holes. For each occluded pixel of the current target billboard BB_t, the location in the best matching billboard IBB_j is given by a displacement d. The displacement is computed by filtering the displacements stored in the grid cells with a Gaussian kernel K centered at the current pixel (line 22). As a result, the displacement d is interpolated from displacements of nearby matched feature pairs, each weighted by the Gaussian kernel. The color and depth channels

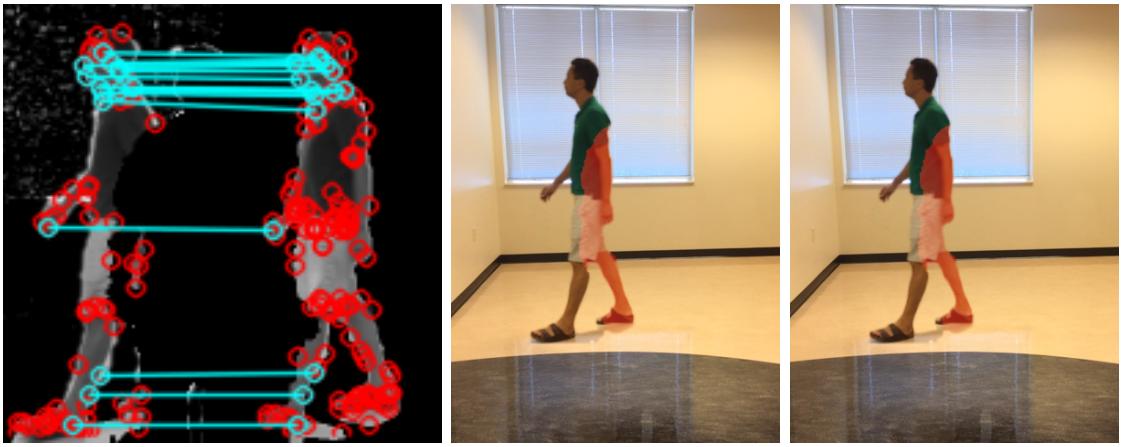


Fig. 5.7.: Left: extracted ORB features. Matched feature pairs are highlighted in cyan. Middle: inpainted billboard without non-rigid alignment. Right: inpainted billboard with non-rigid alignment.

as well as the transparency label of the pixel are set using the corresponding pixel in the cached billboard (lines 23-25). If the corresponding pixel is transparent, i.e. it is a background pixel, the current pixel is also marked as transparent.

With all the missing pixels filled in, Step 4 of the algorithm extracts the complete set of color features from the inpainted target billboard. The billboard, along with extracted features, is inserted into the cache of inpainted billboards for use in future frames.

5.5 Visualization

Once the inpainting stage computes a complete impostor billboard of the target, the target is visualized occlusion free using standard occlusion-removal visualization techniques developed for synthetic scenes such as transparency and cutaway (Fig. 5.8). All objects are rendered on top of the reference frame efficiently as billboards with transparent texels. Correct visibility ordering of the billboards is readily available since the billboards are placed at the correct depth in 3D, leveraging the depth channel, as described above.

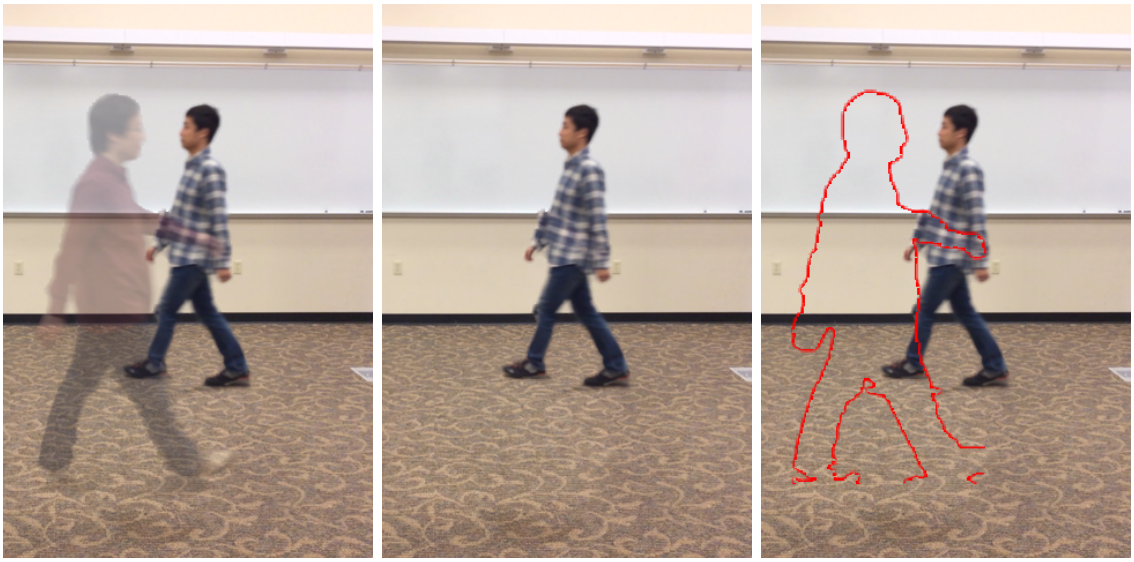


Fig. 5.8.: Video sequence D — Occlusion-free visualizations: transparency (left), cutaway (middle) that completely removes the occluder, and cutaway with occluder silhouette (right).

5.6 Results and Discussion

We tested our system on several video sequences (Fig. 5.1, 5.2, 5.8, and 5.9) captured in both indoor and outdoor settings, with walking humans as targets and occluders. Some sequences have an eye-level viewpoint (e.g. sequences A and D) and some simulate a raised viewpoint like the ones used in surveillance applications (e.g. sequences B and E). In some sequences the target is a human rolling a wheeled bag or pushing a trolley cart, which illustrates that our method does not rely on a walking human model (e.g. sequences B, F, and H). In sequences C, F, G, and H the occluder is synthetic which provides ground truth for the occlusion free frames to which to compare the output of our method (Fig. 5.2).

The acquisition setup consists of a computer tablet (Apple iPad) with an on-board structured light depth camera accessory (Structure Sensor [107]). For our application, the tablet is set to acquire video at a resolution of 640 $\times$ 480 at 30Hz. The depth camera acquires depth at a resolution of 320 $\times$ 240 at 30Hz. The depth camera acquires depth frames in sync with the video frames. The tablet is mounted on a tripod to record RGBD videos from a fixed viewpoint.

5.6.1 Speed

We measured the performance of our pipeline by processing pre-recorded RGBD video sequences on an Intel Xeon E5-1660 3.3GHz workstation with 16GB of memory and with an NVIDIA Quadro K5000 4GB graphics card. The implementation mainly relies on the CPU, but uses OpenGL and GPU shaders written in GLSL for 2D segmentation and final composition. In all our tests, the frame rate is at least 27fps, with an average frame rate of 31.0fps (Table 5.1), or a frame time of around 32.2ms. Out of the three main stages of our pipeline, segmentation and billboard construction (Alg. 1) takes 6ms, and inpainting the target billboard (Alg. 2) takes 21ms. The visualization stage takes up the remaining frame time, i.e. 5ms.

(a) Video sequence E

(b) Video sequence F

(c) Video sequence G

(d) Video sequence H

Fig. 5.9.: Video sequences E to H — Occlusion-free frames generated with our method.

Table 5.1.: Frame rates achieved by our method [fps].

	Video sequence							
	A	B	C	D	E	F	G	H
min	27	28	29	29	28	28	30	29
max	35	38	33	38	37	34	35	37
average	30	31	31	32	31	30	30	33

Within the inpainting stage (Alg. 2), which only processes the target billboard and not the occluder billboards, steps 2 and 4 together take the most time at 13ms, or 40% of the total frame time. The cost arises from the FeatureExtraction function (Alg. 2, lines 11 and 28) which extracts ORB features for the purpose of non-rigid alignment between textures of current and previous target billboards. Performing feature extraction on downsampled billboard textures increases performance at the cost of alignment precision.

We achieve high performance by inpainting from a known region of a carefully selected previous frame. Compared to offline video inpainting methods, which may take image pixels or patches from disjoint regions across multiple frames, our method greatly reduces the search space for inpainting samples. Since our methods respects the inpainting source topology, the cost of the global optimization for the alignment of inpainting samples is also eliminated. Section 5.6.3 discusses the performance comparison in detail.

5.6.2 Quality

As shown in the images in this chapter (Fig. 5.1, 5.2, 5.8, and 5.9) and in the accompanying video, our method produces quality occlusion-free visualizations of the target. Our method does not minimize per pixel error and rather focuses on providing a high-quality (blur free) visualization of the target in a similar pose to the one needed for the current frame. When the target is partially occluded, the inpainting conforms to the parts of the target that are visible. Our system supports a brief total occlusion of the target (Fig. 5.2), case in which the matching billboard is found by assuming a constant time offset to the frame used for inpainting (Alg. 2). Our method has good frame to frame coherence, providing a continuous, occlusion-free visualization of the target.

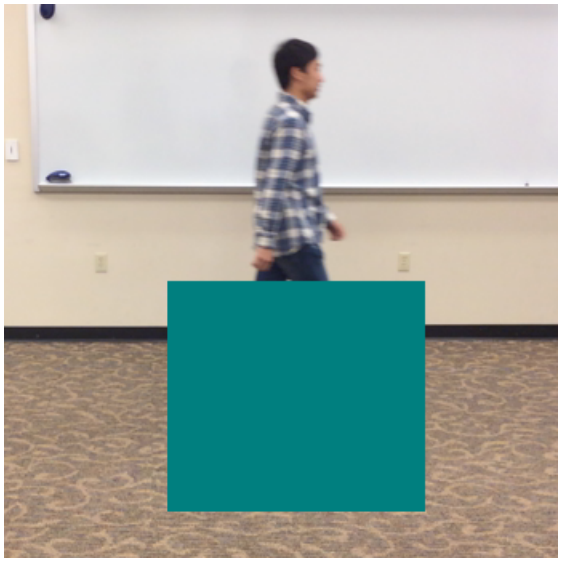
5.6.3 Comparison to prior work

We compare the result of our fast occlusion removal with a state-of-the-art patch-based video inpainting algorithm [100] (Fig. 5.10). To focus the comparison on the deformable and moving target object, we choose one of our video sequences with relatively simple background geometry. A dynamic artificial occluder, which occludes the target only partially, is added to the video sequence, while the original video, without the occluder, serves as ground truth reference. The prior work algorithm takes an additional binary occluder mask video as input.

The prior work method is able to inpaint the occluded parts of the target while maintaining edge continuity. It is also capable of inpainting both the flat and textured parts of the background. However, their method does not produce visually correct results for the target, as evident in the highly deformed legs. On the other hand, our method is able to inpaint the missing parts of the target by borrowing from an earlier frame when the missing parts were visible.

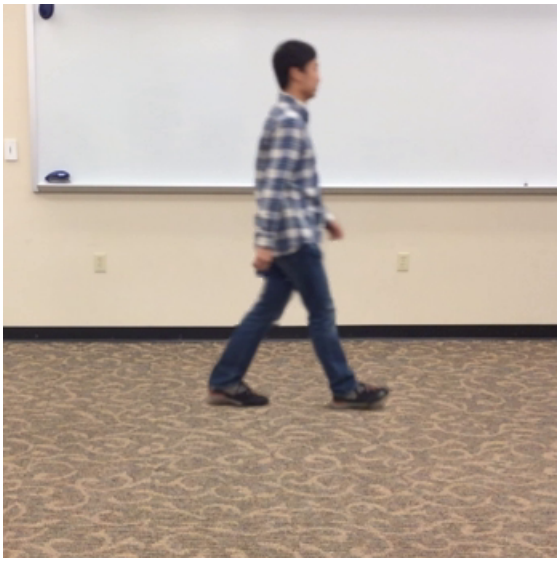
To compare the performance of the two methods algorithmically, we identify the two major operations common to our and prior work algorithms: search and reconstruction. The search operation in both algorithms searches within the video volume for best matching source pixels for inpainting the hole pixels. The reconstruction operation then inpaints the hole by adapting source pixels to the hole region. For the purpose of algorithmic comparison, we denote the number of hole pixels H.

The search operation in our algorithm is efficient because the matching error is only computed in whole between the current billboard and the cached billboard (Alg. 2, Step 1). Furthermore, the matching error is computed only within the search window, w. The complexity is therefore $O(Hw)$. In comparison, the search operation in the prior work algorithm searches for a best matching patch to each hole patch, optimized over many iterations. The overall complexity is $O(k_l H n \mathcal{N}_p)$, where k_l is the number of iterations for optimization, n is the number of source patch candidates, and $\mathcal{N}_p$ is the patch size.



(a) Input

(b) Truth



(c) Our result

(d) Newson et al. 2014 [100]

Fig. 5.10.: Comparison between our method and prior work.

Table 5.2.: Performance comparison

Algorithm	Search	Reconstruction
Ours (predicted)	$O(Hw)$	$O(k_eH + k_mf^2) + T_{GPU}$
Ours (measured)	1.7ms	13ms + 6.6ms
Newson (predicted)	$O(k_lHn\mathcal{N}_p)$	$O(k_lH\mathcal{N}_p^2)$
Newson (measured)	20s	3.1s

The reconstruction operation in our algorithm aligns matched pairs of features extracted from the current and source billboards (Alg. 2, Step 2 and 4). These sparse features are paired by brute force comparison. Overall, these steps incur a complexity of $O(k_eH+k_mf^2)$, where k_e and k_m are constants for feature extraction and matching. Step 3 executes in parallel on the GPU, and the cost is denoted by T_{GPU}. In the prior work algorithm, each hole pixel receives color from $\mathcal{N}_p$ source pixels, where each source pixel's weight is calculated by a patch-to-patch distance of $O(\mathcal{N}_p)$ complexity. The overall complexity is therefore $O(k_lH\mathcal{N}_p^2)$, where k_l is the number of iterations for optimization.

We estimate the constant factors for the complexity of the prior work algorithm with default parameters. Overall, the optimization iteration count k is on the order of 10^2. Furthermore, any pixel-to-pixel complexity is increased to patch-to-patch complexity, where a spatio-temporal patch defaults to a cube of 5^3 pixels. The optimization loop and patch size greatly scale up the run time of the prior work algorithm by a factor of 10^4.

We also compare the actual running times of the two methods. The prior work implementation is obtained from the authors, and it is executed with default parameters. The prior work implementation takes 3,053s to inpaint the 90 frame video segment for an average of 33.9s per frame. Our method takes an average of 32ms per frame for the same video segment, which is a significant performance improvement at 3 orders of magnitude. The prior work algorithm is implemented on the CPU, while certain steps of our pipeline (Alg.1, Step 1 and 3; Alg. 2, Step 3) run on the GPU. A detailed breakdown is provided in Table 5.2.

In order to objectively assess the reconstruction quality, we employ algorithmic metrics to compare the similarity between reconstructed images and truth images. We choose a learning-based metric which calculates the weighted cosine distance between feature vectors extracted from a convolutional encoder [108]. Different from conventional pixel error metrics such as PSNR, learning-based metrics are able to infer high level image structure, which better evaluates inpainting of large regions of missing pixels. The implementation from the authors were used with default parameters for the AlexNet network [109]. Overall, our reconstructed images achieved an average distance of 0.010 ± 0.007 from truth images, significantly less than the prior method which achieved an average distance of 0.021 ± 0.007. Therefore, our method is shown to reconstruct images in the test sequence more similar to the ground truth than the prior method.

5.6.4 Limitations

Limitation in segmentation accuracy leads to artifacts in the impostor billboard textures in several ways: i) The 3D segmentation (Alg. 1, Step 2) can only segment objects separable by bounding cylinders. If the objects are intersecting each other, they cannot be separated cleanly. ii) Background subtraction relies on comparing the current pixel color to the known background color. However, dynamic objects cause lighting changes in the background. iii) The depth channel suffers from noisy or missing depth value, so pixels can be erroneously segmented. Our pipeline can easily leverage any advance in segmentation.

Our method assumes motion coherence over the occlusion time interval, so it does not support abrupt deviations from the motion pattern, such as, for example, the target stopping behind the occluder to look at their watch. Another limitation is the difficulty in predicting target motion when the target is heavily occluded for an extended amount of time. Our method does not rely on a constant motion period and it adapts the time offset to the past frame used for inpainting. However, when the

target is occluded for a long time, this prediction of the position of the target becomes approximate. The difference between the predicted and actual target position can increase until the target reemerges, when the visualization snaps the target back to the correct position and pose.

Our method assumes that the target has the same appearance in the current frame as in the the inpainting source frame. This assumption is violated by a series of view-dependent effects, including parallax between the two frames as the position of the viewpoint relative to the target changes, and shadows and reflections of the target. It is also violated by non-uniform scene lighting which alters the target's appearance between the two frames.

5.7 Conclusions and Future Work

We have implemented a system which removes occlusion to targets in RGBD video streams by target impostor inpainting. Our system operates at interactive rates by leveraging the depth information in several ways: depth keying augments color-based background subtraction; 3D point clouds constructed from depth images support fast object detection and tracking; billboards enhanced with per-texel depth enable efficient billboard to billboard correspondence computation. The system is demonstrated on a variety of real-world scenarios including one or more occluding objects, different camera perspectives, and different object geometries. Performance is greatly improved compared with prior work.

Our current work uses a single RGBD video stream and it shows the disoccluded video from the same reference viewpoint. Future work can explore the use of multiple input RGBD streams: multiple RGBD streams can acquire the target from different viewpoints over longer periods, constructing a comprehensive cache of the target's actions, which potentially increases the range of occlusions that our method can successfully inpaint. To leverage this comprehensive cache requires efficient cache indexing and searching. Inspired by the work of Liu et al. [110], we would like to investigate

efficient search algorithms of best matching frame in the presence of large billboard caches accumulated from multiple video streams. A related possible improvement is to remove the requirement that the tablet be stationary during acquisition, where different approaches to video segmentation are called for.

Our method relies on a cache of previous target billboards, at least one of which needs to be similar to the current, partially occluded billboard. This is challenging for targets that exhibit a wide range of motions. For human targets, we demonstrated our method on various cases of walking motion while interacting with rigid objects. Our method does not require strict periodicity, and can handle acceleration and deceleration. This is an important case, as humans, animals, and birds move fairly uniformly, but with slight variations, over short distances. Objects that move rigidly like cars are much simpler cases.

Recent advances have been made in applying the CNN to image and video inpainting problems [111, 112]. These methods are able to inpaint complex images when the CNNs are trained with a dataset that is representative of the expected input. However, image inpainting using CNN does not straightforwardly preserve temporal coherence when applied to video processing [111], and fast video inpainting using CNN remains challenging due to the computationally costly 3D convolution over the input video volume [112]. We would like to explore potential hybrid approaches that leverage both the efficient billboard caching of our method and the generalized image reconstruction using CNNs.

6. CONCLUSION

This thesis makes the following statement: *Occlusions in desktop and head-mounted display visualization of 3D datasets can be alleviated by relaxing the single viewpoint and single timepoint constraints of conventional images, to increase the information content of the image by integrating dataset samples collected from multiple viewpoints and multiple timepoints.*

In support of our thesis, (1) we presented a multiperspective rendering model that relaxes the single viewpoint constraint, (2) we presented the use of multiperspective rendering in the context of HMD visualization for AR and VR applications, (3) we presented a naturalistic interface for HMD MPV, and (4) we presented an approach for multi-timepoint visualization of real world scenes.

(1) We presented the flexible focus+context camera model that relaxes the single viewpoint constraint. This flexible camera model subdivides the conventional camera frustum into camera segments, each defined as a CGLC. CGLCs in a focus region are assembled into a sub-frustum to image the scene ROI without any distortion. There can be multiple such sub-frusta in a flexible camera model, and each sub-frustum is free to image the region of interest from a novel viewpoint that is disjoint from the main viewpoint. CGLCs in the context regions connect the sub-frusta to provide continuous context visualization. The flexible camera framework is constructed to alleviate occlusions for 3 scenarios: i) top-down exploration, where the user explores the scene using the multiple sub-frusta in parallel, ii) bottom-up integration, where the user is presented with a comprehensive image of multiple known ROIs, and iii) target tracking, where the flexible camera automatically maintains an uninterrupted visualization of the target.

(2) We presented the MPV as an approach to increase navigation efficiency for users of immersive displays in VR and AR. In a first implementation in VR, the user

deploys a secondary perspective that disoccludes the gazed point at ground level, obviating the need for the user to assume different viewpoints in the scene using laborious physical locomotion. In a second implementation in AR, the secondary perspective is deployed for the view portal selected by user gaze, and it seamlessly integrates scene geometry beyond the view portal into the user's primary perspective. By relaxing the single viewpoint constraint in immersive displays, the user is able to selectively disocclude multiple ROIs in VR and AR with minimal physical locomotion. The user study showed significant improvement in navigation efficiency on tasks such as tracking, matching, searching, and ambushing objects of interest.

(3) We presented the Anchored MPV in VR that extends the user's control of the visualization and integrates with teleportation navigation. In this implementation, the visualization integrates an anchoring primary perspective and a controllable secondary perspective. The primary perspective, visualizing the near geometry, is undisturbed and serves to anchor the user in the virtual scene, while the secondary perspective is intuitively oriented by the user's lateral head translation. Furthermore, this implementation supports anchored teleportation, leveraging MPV to provide a continuous visualization of the scene during teleportation. In the user study where the users relied on teleportation for long distance navigation, significant improvements were observed in the number of teleportations and the total distance traveled.

(4) We presented the temporal resampling algorithm that removes occlusion suffered in acquisition using a physical camera by integrating samples from multiple timepoints. We implemented the algorithm in a system that takes an input RGBD video stream, and outputs video visualizations where occlusion to the target of interest is removed. The system first segments the input video stream into foreground objects and the background geometry, leveraging the depth information from the video stream. A textured billboard impostor is created for each foreground object. The billboard impostor for the target is stored in a cache. If the target becomes occluded at one point in time, the occluded region of the target's billboard impostor is inpainted by first selecting a best matching previous billboard from the cache, and

then seamlessly merging the cached samples from the past timepoint with the visible samples from the current timepoint. By carefully selecting matching target impostor billboards, and by resampling from only the specific timepoints, our algorithm greatly reduces the complexity of video inpainting. Due to the novel algorithm, our system achieves interactive frame rates, which enables potential applications of video inpainting in surveillance and diminished reality.

Occlusion arises in complex dataset acquisition or visualization that follow the physical PPC model. By rethinking imaging in graphics and visualization, novel camera models have been designed to alleviate occlusion through careful deviation from the PPC model. In this thesis, we have explored such novel camera models which relax the single viewpoint and single timepoint constraints, and therefore synthesize images that show not only what is visible from the current viewpoint and timepoint, but also what is visible across multiple view and timepoints. The image generalization achieved through this increased freedom in camera model design has been found to increase visual information bandwidth, to improve navigation efficiency in HMDs, and to efficiently infer missing samples in the dataset. Based on the positive results presented, we advocate a departure from conventional rendering algorithms that mimic physical camera models, to explore the expanded design space of camera models to derive eloquent mappings of 3D datasets to 2D images.

REFERENCES

[1] M. L. Wu and V. Popescu, "Multiperspective focus+context visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 5, pp. 1555–1567, May 2016.

[2] ——, "Efficient vr and ar navigation through multiperspective occlusion management," *IEEE Transactions on Visualization and Computer Graphics*, vol. PP, no. 99, pp. 1–1, 2017.

[3] ——, "Anchored multiperspective visualization for efficient vr navigation," in *Virtual Reality and Augmented Reality*, P. Bourdot, S. Cobb, V. Interrante, H. kato, and D. Stricker, Eds. Cham: Springer International Publishing, 2018, pp. 240–259.

[4] ——, "Rgbd temporal resampling for real-time occlusion removal," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '19. New York, NY, USA: ACM, 2019, pp. 7:1–7:9. [Online]. Available: http://doi.acm.org/10.1145/3306131.3317025

[5] P. Rosen and V. Popescu, "An evaluation of 3-d scene exploration using a multiperspective image framework," *The Visual Computer*, vol. 27, no. 6, pp. 623–632, Jun 2011. [Online]. Available: https://doi.org/10.1007/s00371-011-0599-2

[6] N. Elmqvist and P. Tsigas, "A taxonomy of 3d occlusion management for visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 5, pp. 1095–1109, Sep. 2008. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2008.59

[7] N. Wong, S. Carpendale, and S. Greenberg, "Edgelens: an interactive method for managing edge congestion in graphs," in *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No.03TH8714)*, Oct 2003, pp. 51–58.

[8] P. Degener, R. Schnabel, C. Schwartz, and R. Klein, "Effective visualization of short routes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1452–1458, Nov 2008.

[9] S. Möser, P. Degener, R. Wahl, and R. Klein, "Context aware terrain visualization for wayfinding and navigation," *Computer Graphics Forum*, vol. 27, no. 7, pp. 1853–1860, Oct. 2008.

[10] S. Takahashi, K. Yoshida, K. Shimada, and T. Nishita, "Occlusion-free animation of driving routes for car navigation systems," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1141–1148, Sept 2006.

[11] Y. Kurzion and R. Yagel, "Space deformation using ray deflectors," in *Rendering Techniques '95*, P. M. Hanrahan and W. Purgathofer, Eds. Vienna: Springer Vienna, 1995, pp. 21–30.

[12] M. Agrawala, D. Zorin, and T. Munzner, "Artistic multiprojection rendering," in *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*. London, UK, UK: Springer-Verlag, 2000, pp. 125–136. [Online]. Available: http://dl.acm.org/citation.cfm?id=647652.732127

[13] P. Degener and R. Klein, "A variational approach for automatic generation of panoramic maps," *ACM Trans. Graph.*, vol. 28, no. 1, pp. 2:1–2:14, Feb. 2009. [Online]. Available: http://doi.acm.org/10.1145/1477926.1477928

[14] M. Falk, T. Schafhitzel, D. Weiskopf, and T. Ertl, "Panorama maps with non-linear ray tracing," in *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*, ser. GRAPHITE '07. New York, NY, USA: ACM, 2007, pp. 9–16. [Online]. Available: http://doi.acm.org/10.1145/1321261.1321263

[15] H. Jenny, B. Jenny, W. E. Cartwright, and L. Hurni, "Interactive local terrain deformation inspired by hand-painted panoramas," *The Cartographic Journal*, vol. 48, no. 1, pp. 11–20, 2011. [Online]. Available: https://doi.org/10.1179/1743277411Y.0000000002

[16] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski, "Photographing long scenes with multi-viewpoint panoramas," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 853–861, Jul. 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1141966

[17] A. Roman, G. Garg, and M. Levoy, "Interactive design of multi-perspective images for visualizing urban landscapes," in *IEEE Visualization 2004*, Oct 2004, pp. 537–544.

[18] D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. H. Salesin, "Multiperspective panoramas for cel animation," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1997, pp. 243–250. [Online]. Available: https://doi.org/10.1145/258734.258859

[19] P. Rademacher and G. Bishop, "Multiple-center-of-projection images," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 199–206.

[20] J. Yu and L. McMillan, "General linear cameras," in *Computer Vision - ECCV 2004*, T. Pajdla and J. Matas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 14–27.

[21] ——, "A framework for multiperspective rendering," in *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, ser. EGSR'04. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2004, pp. 61–68. [Online]. Available: http://dx.doi.org/10.2312/EGWR/EGSR04/061-068

[22] V. Popescu, J. Dauble, C. Mei, and E. Sacks, "An efficient error-bounded general camera model," in *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, June 2006, pp. 121–128.

[23] C. Mei, V. Popescu, and E. Sacks, "The Occlusion Camera," *Computer Graphics Forum*, 2005.

[24] V. Popescu, P. Rosen, and N. Adamo-Villani, "The graph camera," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 158:1–158:8, Dec. 2009.

[25] C. Correa, D. Silver, and M. Chen, "Illustrative deformation for data exploration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1320–1327, Nov 2007.

[26] E. Gröller, "Nonlinear ray tracing: Visualizing strange worlds," *The Visual Computer*, vol. 11, no. 5, pp. 263–274, May 1995. [Online]. Available: https://doi.org/10.1007/BF01901044

[27] Y. Kurzion and R. Yagel, "Interactive space deformation with hardware-assisted rendering," *IEEE Computer Graphics and Applications*, vol. 17, no. 5, pp. 66–77, Sep 1997.

[28] D. Weiskopf, T. Schafhitzel, and T. Ertl, "Gpu-based nonlinear ray tracing," vol. 23, pp. 625–634, 09 2004.

[29] J. Cui, P. Rosen, V. Popescu, and C. Hoffmann, "A curved ray camera for handling occlusions through continuous multiperspective visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1235–1242, Nov 2010.

[30] C. D. Correa and D. Silver, "Programmable shaders for deformation rendering," in *Proceedings of the 22Nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware*, ser. GH '07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 89–96. [Online]. Available: http://dl.acm.org/citation.cfm?id=1280094.1280109

[31] M. Hadwiger, P. Ljung, C. R. Salama, and T. Ropinski, "Advanced illumination techniques for gpu-based volume raycasting," in *ACM SIGGRAPH 2009 Courses*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 2:1–2:166. [Online]. Available: http://doi.acm.org/10.1145/1667239.1667241

[32] W. R. Mark, R. S. Glanville, K. Akeley, and M. J. Kilgard, "Cg: A system for programming graphics hardware in a c-like language," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 896–907, Jul. 2003. [Online]. Available: http://doi.acm.org/10.1145/882262.882362

[33] S. Bruckner and M. E. Groller, "Exploded views for volume data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1077–1084, Sept 2006.

[34] W. Li, M. Agrawala, B. Curless, and D. Salesin, "Automated generation of interactive 3d exploded view diagrams," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 101:1–101:7, Aug. 2008.

[35] M. Burns and A. Finkelstein, "Adaptive cutaways for comprehensible rendering of polygonal scenes," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 154:1–154:7, Dec. 2008.

[36] J. Kruger, J. Schneider, and R. Westermann, "Clearview: An interactive context preserving hotspot visualization technique," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 941–948, Sept 2006.

[37] R. J. C. Hilf, C. Bertozzi, I. Zimmermann, A. Reiter, D. Trauner, and R. Dutzler, "Structural basis of open channel block in a prokaryotic pentameric ligand-gated ion channel," *Nature Structural & Molecular Biology*, vol. 17, pp. 1330–1336, October 2010. [Online]. Available: http://dx.doi.org/10.1038/nsmb.1933

[38] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The protein data bank," *Nucleic Acids Research*, vol. 28, no. 1, pp. 235–242, 2000. [Online]. Available: http://dx.doi.org/10.1093/nar/28.1.235

[39] R. Reijerse, 2006. [Online]. Available: http://reije081.home.xs4all.nl/skyboxes/

[40] J. L. Souman, P. R. Giordano, M. Schwaiger, I. Frissen, T. Thümmel, H. Ulbrich, A. D. Luca, H. H. Bülthoff, and M. O. Ernst, "Cyberwalk: Enabling unconstrained omnidirectional walking through virtual environments," *ACM Trans. Appl. Percept.*, vol. 8, no. 4, pp. 25:1–25:22, Dec. 2008.

[41] S. Tregillus and E. Folmer, "Vr-step: Walking-in-place using inertial sensing for hands free navigation in mobile vr environments," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: ACM, 2016, pp. 1250–1255.

[42] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F. P. Brooks, Jr., "Walking > walking-in-place > flying, in virtual environments," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '99. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 359–364.

[43] R. A. Ruddle and S. Lessels, "The benefits of using a walking interface to navigate virtual environments," *ACM Trans. Comput.-Hum. Interact.*, vol. 16, no. 1, pp. 5:1–5:18, Apr. 2009.

[44] S. Razzaque, D. Swapp, M. Slater, M. C. Whitton, and A. Steed, "Redirected walking in place," in *Proceedings of the Workshop on Virtual Environments 2002*, ser. EGVE '02. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2002, pp. 123–130.

[45] S. Razzaque, Z. Kohn, and M. C. Whitton, "Redirected Walking," in *Eurographics 2001 - Short Presentations*. Eurographics Association, 2001.

[46] J. J. LaViola, Jr., D. A. Feliz, D. F. Keefe, and R. C. Zeleznik, "Hands-free multi-scale navigation in virtual environments," in *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, ser. I3D '01. New York, NY, USA: ACM, 2001, pp. 9–15.

[47] X. Xie, Q. Lin, H. Wu, G. Narasimham, T. P. McNamara, J. Rieser, and B. Bodenheimer, "A system for exploring large virtual environments that combines scaled translational gain and interventions," in *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization*, ser. APGV '10. New York, NY, USA: ACM, 2010, pp. 65–72.

[48] B. Williams, G. Narasimham, B. Rump, T. P. McNamara, T. H. Carr, J. Rieser, and B. Bodenheimer, "Exploring large virtual environments with an hmd when physical space is limited," in *Proceedings of the 4th Symposium on Applied Perception in Graphics and Visualization*, ser. APGV '07. New York, NY, USA: ACM, 2007, pp. 41–48.

[49] Q. Sun, L.-Y. Wei, and A. Kaufman, "Mapping virtual and physical reality," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 64:1–64:12, Jul. 2016.

[50] F. Steinicke, G. Bruder, J. Jerald, H. Frenz, and M. Lappe, "Analyses of human sensitivity to redirected walking," in *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, ser. VRST '08. New York, NY, USA: ACM, 2008, pp. 149–156.

[51] ——, "Estimation of detection thresholds for redirected walking techniques," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 1, pp. 17–27, Jan 2010.

[52] Y. Kameda, T. Takemasa, and Y. Ohta, "Outdoor see-through vision utilizing surveillance cameras," in *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov 2004, pp. 151–160.

[53] B. Avery, C. Sandor, and B. H. Thomas, "Improving spatial perception for augmented reality x-ray vision," in *2009 IEEE Virtual Reality Conference*, March 2009, pp. 79–82.

[54] C. Sandor, A. Cunningham, A. Dey, and V. V. Mattila, "An augmented reality x-ray system based on visual saliency," in *2010 IEEE International Symposium on Mixed and Augmented Reality*, Oct 2010, pp. 27–36.

[55] S. Zollmann, D. Kalkofen, E. Mendez, and G. Reitmayr, "Image-based ghostings for single layer occlusions in augmented reality," in *2010 IEEE International Symposium on Mixed and Augmented Reality*, Oct 2010, pp. 19–26.

[56] S. Zollmann, R. Grasset, G. Reitmayr, and T. Langlotz, "Image-based x-ray visualization techniques for spatial understanding in outdoor augmented reality," in *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design*, ser. OzCHI '14. New York, NY, USA: ACM, 2014, pp. 194–203.

[57] W. Li, L. Ritter, M. Agrawala, B. Curless, and D. Salesin, "Interactive cutaway illustrations of complex 3d models," *ACM Trans. Graph.*, vol. 26, no. 3, Jul. 2007.

[58] D. Kalkofen, M. Tatzgern, and D. Schmalstieg, "Explosion diagrams in augmented reality," in *2009 IEEE Virtual Reality Conference*, March 2009, pp. 71–78.

[59] H. Deng, L. Zhang, X. Mao, and H. Qu, "Interactive urban context-aware visualization via multiple disocclusion operators," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 7, pp. 1862–1874, July 2016.

[60] R. I. Hartley and R. Gupta, *Linear pushbroom cameras.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 555–566.

[61] J. Yu and L. McMillan, *General Linear Cameras.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 14–27.

[62] E. Veas, R. Grasset, E. Kruijff, and D. Schmalstieg, "Extended overview techniques for outdoor augmented reality," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 4, pp. 565–572, April 2012.

[63] H. Lorenz, M. Trapp, J. Döllner, and M. Jobst, *Interactive Multi-Perspective Views of Virtual 3D Landscape and City Models.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 301–321.

[64] S. Kim and A. K. Dey, "Simulated augmented reality windshield display as a cognitive mapping aid for elder driver navigation," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '09. New York, NY, USA: ACM, 2009, pp. 133–142.

[65] M. Tatzgern, D. Kalkofen, R. Grasset, and D. Schmalstieg, "Embedded virtual views for augmented reality navigation," *International Symposium On Mixed and Augmented Reality - Workshop on Visualization in Mixed Reality Environments*, 2011.

[66] C. Sandor, A. Dey, A. Cunningham, S. Barbier, U. Eck, D. Urquhart, M. R. Marner, G. Jarvis, and S. Rhee, "Egocentric space-distorting visualizations for rapid environment exploration in mobile mixed reality," in *2010 IEEE Virtual Reality Conference (VR)*, March 2010, pp. 47–50.

[67] J. Cohen, *Statistical power analysis for the behavioral sciences: Jacob Cohen.* Psychology Press, 2009.

[68] S. S. Sawilowsky, "New effect size rules of thumb," *Journal of Modern Applied Statistical Methods*, vol. 8, no. 2, pp. 597–599, 2009.

[69] E. Bozgeyikli, A. Raij, S. Katkoori, and R. Dubey, "Point & teleport locomotion technique for virtual reality," in *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, ser. CHI PLAY '16. New York, NY, USA: ACM, 2016, pp. 205–216. [Online]. Available: http://doi.acm.org/10.1145/2967934.2968105

[70] J. J. Laviola, "A discussion of cybersickness in virtual environments," *SIGCHI Bulletin*, 2000.

[71] S. Davis, K. Nesbitt, and E. Nalivaiko, "A systematic review of cybersickness," in *Proceedings of the 2014 Conference on Interactive Entertainment*, ser. IE2014. New York, NY, USA: ACM, 2014, pp. 8:1–8:9. [Online]. Available: http://doi.acm.org/10.1145/2677758.2677780

[72] J. Habgood, D. Moore, D. Wilson, and S. Alapont, "Rapid, continuous movement between nodes as an accessible virtual reality locomotion technique," in *Proceedings of the 25th IEEE Conference on Virtual Reality and 3D User Interfaces*, ser. VR '18, Reutlingen, Germany, 2018.

[73] E. D. Ragan, S. Scerbo, F. Bacim, and D. A. Bowman, "Amplified head rotation in virtual reality and the effects on 3d search, training transfer, and spatial orientation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 8, pp. 1880–1895, Aug 2017.

[74] A. Kunert, A. Kulik, S. Beck, and B. Froehlich, "Photoportals: Shared references in space and time," in *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, ser. CSCW '14. New York, NY, USA: ACM, 2014, pp. 1388–1399.

[75] Neat Corporation, "Budget cuts," 2018. [Online]. Available: http://neatcorporation.com/

[76] D. R. Montello, "Scale and multiple psychologies of space," in *Spatial Information Theory A Theoretical Basis for GIS*, A. U. Frank and I. Campari, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 312–321.

[77] T. Weissker, A. Kunert, B. Froehlich, and A. Kulik, "Spatial updating and simulator sickness during steering and jumping in immersive virtual environments," in *Proceedings of the 25th IEEE Conference on Virtual Reality and 3D User Interfaces*, ser. VR '18, Reutlingen, Germany, 2018.

[78] Microsoft, "Windows mixed reality." [Online]. Available: http://www.microsoft.com/

[79] id Software, "Quake 3 arena," 1999.

[80] L. Joyeux, O. Buisson, B. Besserer, and S. Boukir, "Detection and removal of line scratches in motion picture films," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1, 1999, p. 553 Vol. 1.

[81] A. C. Kokaram, "On missing data treatment for degraded video and film archives: a survey and a new bayesian approach," *IEEE Transactions on Image Processing*, vol. 13, no. 3, pp. 397–415, March 2004.

[82] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[83] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *In International Conference on Computer Vision*, 1999, pp. 1033–1038.

[84] A. Bugeau, M. Bertalmio, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2634–2645, Oct 2010.

[85] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," in *ACM SIGGRAPH 2009 Papers*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 24:1–24:11.

[86] C. Barnes and F.-L. Zhang, "A survey of the state-of-the-art in patch-based synthesis," *Computational Visual Media*, vol. 3, no. 1, pp. 3–20, Mar 2017. [Online]. Available: https://doi.org/10.1007/s41095-016-0064-2

[87] M. M. Oliveira, B. Bowen, R. McKenna, and Y.-S. Chang, "Fast digital image inpainting," in *Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain*, 2001, pp. 106–107.

[88] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 861–868, Jul. 2005.

[89] M. Elad, J.-L. Starck, P. Querre, and D. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (mca)," *Applied and Computational Harmonic Analysis*, vol. 19, no. 3, pp. 340 – 358, 2005.

[90] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 882–889, Aug 2003.

[91] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1200–1212, Sept 2004.

[92] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 463–476, March 2007.

[93] T. Shiratori, Y. Matsushita, X. Tang, and S. B. Kang, "Video completion by motion field transfer," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, June 2006, pp. 411–418.

[94] M. Liu, S. Chen, J. Liu, and X. Tang, "Video completion via motion guided spatial-temporal global optimization," in *Proceedings of the 17th ACM International Conference on Multimedia*, ser. MM '09. New York, NY, USA: ACM, 2009, pp. 537–540.

[95] J. Herling and W. Broll, "High-quality real-time video inpainting with pixmix," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 6, pp. 866–879, June 2014.

[96] N. Kawai, T. Sato, and N. Yokoya, "Diminished reality based on image inpainting considering background geometry," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 3, pp. 1236–1247, March 2016.

[97] T. Xue, M. Rubinstein, C. Liu, and W. T. Freeman, "A computational approach for obstruction-free photography," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 34, no. 4, 2015.

[98] F. Klose, O. Wang, J.-C. Bazin, M. Magnor, and A. Sorkine-Hornung, "Sampling based scene-space video processing," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 67:1–67:11, Jul. 2015.

[99] M. Granados, J. Tompkin, K. Kim, O. Grau, J. Kautz, and C. Theobalt, "How not to be seen—object removal from videos of crowded scenes," in *Computer Graphics Forum*, vol. 31, no. 2pt1. Wiley Online Library, 2012, pp. 219–228.

[100] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Prez, "Video inpainting of complex scenes," *SIAM Journal on Imaging Sciences*, vol. 7, no. 4, pp. 1993–2019, 2014.

[101] Y.-T. Jia, S.-M. Hu, and R. R. Martin, "Video completion using tracking and fragment merging," *The Visual Computer*, vol. 21, no. 8, pp. 601–610, 2005.

[102] S. C. S. Cheung, J. Zhao, and M. V. Venkatesh, "Efficient object-based video inpainting," in *2006 International Conference on Image Processing*, Oct 2006, pp. 705–708.

[103] C. H. Ling, C. W. Lin, C. W. Su, Y. S. Chen, and H. Y. M. Liao, "Virtual contour guided video object inpainting using posture mapping and retrieval," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 292–302, April 2011.

[104] J. Huang and X. Tang, "A fast video inpainting algorithm based on state matching," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Oct 2016, pp. 114–118.

[105] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[106] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 2564–2571.

[107] Occipital, "Structure sensor." [Online]. Available: http://structure.io/

[108] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[109] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[110] J. Liu, N. Akhtar, and A. Mian, "Viewpoint invariant rgb-d human action recognition," in *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, Nov 2017, pp. 1–8.

[111] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[112] C. Wang, H. Huang, X. Han, and J. Wang, "Video inpainting by jointly learning temporal structure and spatial details." in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

VITA

Meng-Lin Wu is a Ph.D. student in the Computer Graphics and Visualization Lab (CGVLAB) at Purdue University. He received B.S. and M.S. degrees in physics from National Taiwan University while participating in experimental high energy physics at KEK, Japan. Prior to joining Purdue, he developed game physics at International Games System.

His research is concerned with occlusion and visibility management in data visualization and mixed reality through novel sampling models. Specifically, he is interested in the approach of multi-viewpoint/timepoint rendering using non-linear sampling rays and non-uniform sampling rates.