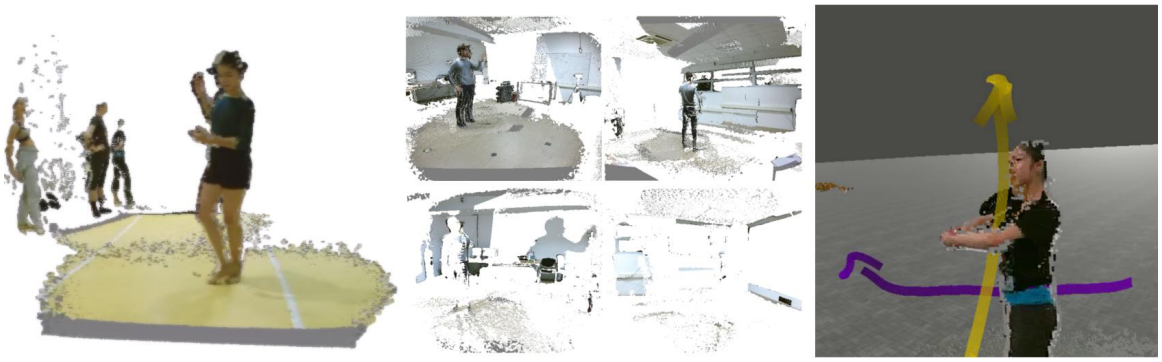
UNIVERSIDADE DE LISBOA

INSTITUTO SUPERIOR TÉCNICO



Multiview Layered Depth Image:
A Data Representation for 3D Flashback

Rafael Kuffner dos Anjos

Orientador: Doctor João António Madeiras Pereira
Co-Supervisors: Doctor José António da Cruz Pinto Gaspar
Doctor Carla Montez Fernandes

Thesis approved in public session to obtain the PhD Degree in Information Systems and Computer Engineering

Jury final classification: Pass with Distinction

2018

UNIVERSIDADE DE LISBOA

INSTITUTO SUPERIOR TÉCNICO

Multiview Layered Depth Image:
A Data Representation for 3D Flashback

Rafael Kuffner dos Anjos

Supervisor: Doctor João António Madeiras Pereira
Co-Supervisors: Doctor José António da Cruz Pinto Gaspar
Doctor Carla Montez Fernandes

Thesis approved in public session to obtain the PhD Degree in
Information Systems and Computer Engineering

Jury final classification: Pass with Distinction

Jury

Chairperson: Doutor Mário Jorge Costa Gaspar da Silva, Instituto Superior Técnico, Universidade de Lisboa

Members of the Comittee:

Doctor Nuno Manuel Robalo Correia, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa.

Doctor António Augusto de Sousa, Faculdade de Engenharia, Universidade do Porto.

Doctor João António Madeiras Pereira, Instituto Superior Técnico, Universidade de Lisboa.

Doctor Carlos António Roque Martinho, Instituto Superior Técnico, Universidade de Lisboa.

Funding Institutions
European Research Council (Ref. 336200)

2018

Resumo

A reconstrução tri-dimensional de eventos capturados em imagem, e a sua representação a partir de um ponto de vista escolhido pelo utilizador, são desafios nas áreas de computação gráfica e visão computacional. Ao lidar com dados de vídeo, esse problema é intitulado "Síntese de gráficos baseada em vídeo". As técnicas propostas em sistemas de televisão com ponto de vista livre possibilitam o movimento da câmera somente através de pontos de vista previamente definidos; consequentemente não contemplando a estrutura tri-dimensional e temporal dos componentes reconstruídos. Esta tese pretende avançar a investigação e o desenvolvimento de técnicas que permitam que um utilizador possa assistir a um evento a partir de um ponto de vista à sua escolha em tempo real, numa reconstrução tri-dimensional detalhada da realidade capturada em vídeo. A reconstrução tri-dimensional de dados de vídeo e a sua codificação numa representação eficiente para posterior reprodução, continuam a ser problemas de investigação por resolver. Um sistema completo foi desenvolvido para responder estes problemas, chamado de "Sistema de Flashback 3D". Três diferentes aplicações no domínio das artes performativas e dança foram desenvolvidas para validar a aproximação proposta.

A principal contribuição desta tese de doutoramento é uma nova representação de dados para vídeos de nuvens de pontos, que também pode ser usada em cenários complexos de síntese de gráficos baseada em imagem; "Multiview Layered Depth Image".

Abstract

Three-dimensional reconstruction of events recorded on images and their representation from a user-chosen viewpoint are a challenge in the fields of computer graphics and computer vision. When dealing with video data, this problem is called video-based rendering. The techniques proposed on free viewpoint television systems enable camera movement based only on the prerecorded viewpoints, thus not properly contemplating the 3D structure and temporal component reconstruction. This thesis aims to go further on research and development of techniques that allows users to watch events and select the desired viewpoint in real time of a detailed 3D reconstruction of the video-captured reality. Three-dimensional reconstruction of captured data on a temporal sequence and its codification on an efficient representation for posterior reproduction still poses as a research challenge to be solved. A complete framework was developed to address these issues, named the 3D flashback framework. Three different applications in the field of performance arts and dance were developed to validate the proposed approach.

The main contribution of this PhD Thesis is a novel data-representation for point cloud videos, which can also be used in complex image-based rendering scenarios, the Multiview Layered Depth Image.

Palavras Chave

Síntese de gráficos baseada em vídeo
Representações baseadas em imagem
Nuvens de pontos
Técnicas de síntese de gráficos
Aplicações interactivas

Keywords

Video-based rendering
Image-based representations
Point-clouds
Rendering techniques
Interactive applications

Acknowledgements

Firstly, I must be thankful to God. This has been a long journey, and through the toughest times I have received His strength and felt His peace. He is the one who has guided me until here, and who will guide me through the future years. Also, I cannot thank my family enough. Thanks dad, mom, and sis. Thank you for supporting me in this decision of studying five more years when you thought I was done with it. Thank you for supporting me financially when I didn't have a scholarship, and thank you for your overwhelming love that I feel everyday.

A special thank you to my wife Letícia, who courageously hopped on the "date a PhD student" train. Thanks for listening to me, comforting me, cheering me up, taking me out, making me smile, and thank you for choosing to be with me for the rest of our lives. I love you!

Also, I want to thank my supervisors Prof. João Pereira for being always ready to answer a phone-call and discuss work, Prof. José Gaspar for guiding me into the world of computer vision, Prof. Carla Fernandes for helping me think "interdisciplinarily".

I want to thank all the people in Vimmi who somehow collaborated with my work. And a more specific thanks to the ones with whom I have either collaborated, shared a room, or ate unhealthy food from a vending machine. Daniel, Mauricio, Daniel, Soraia, Filipe, and one more Daniel.

Thank you to my colleagues in the BlackBox project, Claudia, Vito, Joanna, Stephan, Silvia and Liz.

Thanks to NVIDIA for providing me with a great graphics card to work with. Official thanks to the different funding bodies: European Research Council (project Ref. 336200). FCT (project PEst-OE/EEI/LA0021/2013 and reference UID/CEC/50021 /2013).

Finally, huge huge thanks to the cleaning staff at Tagus Park for keeping our rooms clean and our trash bins empty. I couldn't have done this without you.

"I can do all things through Christ which strengtheneth me"
Phillipians-4:13

.

Contents

List of Figures

List of Tables

Glossary

3DTV three-dimensional television. 1, 2, 42

ABC Artificial Bee Colony. 20
AR augmented reality. 119

CM covariance matrix. 6, 92, 93, 101, 102
CTR Composição em Tempo Real. 4, 117

DIBR depth image-based rendering. xv, 27, 28, 30, 32–34, 36

EB Eberly. 92, 93

HH Householder. 6, 91–93
HMD head mounted display. 119, 120

IBR image-based rendering. 6, 13, 15
ISNPR Image-Space Non-Photorealistic Rendering. 87, 93, 94, 99, 100

LDI Layered Depth Image. xvi, xix, 5, 30, 36, 42–45, 47–54, 66, 68, 69
LDV Layered Depth Video. 30

MPEG Moving Picture Experts Group. 20
MVD Multiview+Depth. 5, 30, 36, 42–45, 51, 53, 68
MVLDI Multiview Layered Depth Image. xvi, xix, 5–7, 42, 43, 45, 46, 49–56, 58, 60–62, 66, 68, 69, 72, 111, 116
MVLDV Multiview Layered Depth Video. xii, xviii, 72, 108, 109
MVS multiview stereo. xii, 20, 21, 26, 27, 30, 35

NPR Non-photorealistic rendering. 88, 89
NTP Network Time Protocol. 78, 79

SAS surface aligned splatting. 87
SBR stroke-based rendering. 93, 94
SBS stroke-based splatting. xvii, 87, 90, 98, 103–105
SfM structure from motion. xv, 20–22, 27, 35

SIFT Scale Invariant Feature Transforms. 17, 18
SQP square plate. 92, 93
SURF Speeded-Up Robust Features. 18

VBR video-based rendering. xii, xv, xvii, 1, 2, 4–16, 22–29, 31–33, 35, 40, 42, 44, 55, 56, 67, 69, 71, 76, 80, 109, 142
VI view interpolation. 32, 34, 35
VR virtual reality. 119–121, 123, 124

1 Introduction

For a long time video has been used in our daily lives as the media that more closely recreates an event as we live it in the real world. The recent popularization of personal video cameras and video content distribution has been pushing the scientific community to expand the traditional video format beyond its classical restrictions such as reproduction speed, which gave birth to slow motion videos, and most recently the viewpoint restriction. The process that uses video as input in order to create novel rendered content is generally defined as video-based rendering (VBR). This field shares goals and challenges with Image-based rendering, while having the extra time dimension that is non-existent in its counterpart. By analyzing the visual content of these images, one tries to extract enough data to add processed information to the existing content or to create novel views that extrapolate the original experience.

The most popular goal on the field is known as three-dimensional television (3DTV)). Not to be confused with stereoscopic displays that are commercially available nowadays, 3DTV aims to allow the user to navigate three-dimensionally around the watched point of view, transforming the content to be correctly displayed accordingly to the new perspective. One early public example of the potential of this technique was the EyeVision (Kitahara et al., 2001) setup used on the Super Bowl XXXV. New viewpoints were not generated by this approach, but the user would get

the natural feel of rotating around an object going from one camera view to another. The field has developed in this direction (Kilner et al., 2007)(Goorts et al., 2013) but without achieving the final goal of 3DTV with no positional restrictions.

When a complete 3D reconstruction of a recorded event is performed, this viewpoint restriction ceases to be a problem given the fact that we have enough information to render the scene from any arbitrary viewpoint. This type of media which we call three-dimensional video can be useful in several types of applications such as security, immersive teleconference and in the chosen test case scenario for this thesis; recording of dance performances. The recent appearing of specified acquisition and visualization hardware allows for an easier capturing process of this data. Without the need for disparity estimation more capturing setups are possible and different types of applications can be developed.

Although advances on reconstruction processes and visualization need to be made, the core limitation for this type of data is the data representation. Currently there is no clear cut best data representation format for video-based rendering (VBR). With such a large array of different applications to fall under this definition and each one having very different requirements, several different approaches have been taken. However, none of them perform efficiently on a wide baseline setup three-dimensional video, which is the goal of this thesis proposal

Our objective is the development of a process that creates this type of media with video-based rendering applications in sight, focusing on an efficient data representation that allows us to capture encode and reproduce such content seamlessly. In this document we refer to this process as the **3D Flashback framework**, described in Section 4.1

1.1. Motivation: BlackBox Project

BlackBox [1] is an interdisciplinary project between arts, cognition, and computer science hosted at FCSH-UNL and led by Prof. Carla Fernandes, in the context of which this PhD thesis is being developed. This project aims to develop a model for a web-based collaborative platform dedicated to documenting the compositional process used by choreographers on contemporary dance and theatre.

Dance and other performing arts are classically taught by example or scores. How-

[1] BlackBox- Arts and Cognition: http://blackbox.fcsh.unl.pt/

ever, creating a defined vocabulary of all possible moves on certain emergent dance movements is an ill posed task, due to the increased creative liberty that the choreographers have. Without a limited number of possible steps, scoring becomes impossible. In these scenarios, if one cannot be taught by example personally, video recordings of performances and rehearsals have been used. Also, for the sake of knowledge preservation and cultural heritage, lengthy video documentation processes have been performed in order to register the more intricate aspects of certain works or movements.

Even when enhancing these videos with contextualized annotations (Wittenburg et al., 2006; Kipp, 2010; ChoreoPro, 2014), the loss of the third dimension and the single-viewpoint limitation is greatly felt. The sense of space and proximity between objects is perceived differently through video when compared to the real world, depending on framing, field of view, and placement of the captured subjects. Moreover, two-dimensional videos are prone to suffer from occlusions, and even when multiview captures are used, mentally combining different viewpoints to recreate a movement in its entirety is not an easy task.

Current approaches for capture and documentation are not capable of properly registering a performance for later visualization, and the same problems apply to the creative process. A great quantity of work is done before a play is presented to the public, and documenting such information is not efficient with the current means which either lose the visual impact of a video, or the structural coherence of a 3D based system.

Allowing one to three-dimensionally capture an event such as a dance performance, process and enhance with meta-information is not only a great documenting tool but can also be a powerful teaching tool. Our proposed wide-baseline 3D Flashback framework allows one to freely navigate three-dimensional data, correctly perceiving the three-dimensional space and relationship between different subjects and objects in a piece. Also, combining the different capture streams into a unified 3D representation takes us one step closer to the *in-loco* experience that has been the standard for teaching dance and performing arts.

Specially in this scenario lightweight representation is crucial. When performances can be an hour long, and include several moving elements at the same time, using a naive data representation makes it impossible to not only store such data, but also integrate it within interactive applications.

During the development of this thesis, we worked with two different contemporary Portuguese choreographers, accompanying the creation of their most recent work.

With João Fiadeiro, who works with conceptual dance, we focused on following workshops related to his composition method (CTR: Composição em Tempo Real, Section 5.3), where we were able to have more insight on the pre-stage work. We recorded two different sessions that lasted approximately three hours each, where shorter improvisation sessions of approximately 20 minutes each would happen with breaks for discussion in between. Part of the data captured in these sessions was used for the work presented in Sections 5.3 and 5.1. The data captured in these section is geometrically complex and naturally creates a lot of occlusions due to the fact that the performers use several props and objects while dancing. Temporally they are simpler, since the performers move slowly.

More recently, we followed the creation of the play "Quinze Bailarinos e Tempo Incerto" by Rui Lopes Graça, a contemporary choreographer usually referred to as neo-classical. Together with the rest of the BlackBox team, we observed and recorded the creative process from the beginning to the end following specifically two dancers: Miyu Matsui and Killian Souc. We captured data during the rehearsals where big groups were visible, but due to the fact we were constrained to where we could place the sensors, we scheduled specific capture sessions where we recorded minute-long solo or duo performances of their material from this play. This data was used for the work described in Sections 3.2, 3.3 and 5.2. While the data is geometrically simpler in a single frame, due to the fact that they contain mostly one or two performers that can be clearly segmented, it is temporally complex since they are constantly moving at high speed.

We applied our proposed solution to different use-case scenarios related to dance and performance arts, as described in Chapter 5.

1.2. Goals

The process of creating three-dimensional videos can be segmented in three stages: **reconstruction**, **representation**, and **visualization**. We believe the key factor to create an efficient VBR system lies on the chosen data representation, which allows us to encode sufficient information to visualize a scene from an arbitrary point of view while having minimal loss of data.

Based on previous work on data representation for VBR (Section 2.5.2) we aim to develop a system that captures data using depth plus color cameras, creates an adaptation of the Layered-depth video data format, and renders it as a sequence of

point clouds on an interactive system. This choice of representation will be discussed in Section 4.2 Although our main research focus will be on the data representation, a system that correctly creates and uses it are essential to prove the viability of the chosen representation for a VBR system.

Our research hypothesis arguments that creating layered depth videos by warping different layers of three-dimensional data into chosen viewpoints will allow us to achieve higher compression rates, while maintaining visual quality. The best suited viewpoint will be chosen according to an efficiency criteria. Developing such data representation and choosing the correct viewpoints is the main focus of this PhD thesis.

Our main research goals can be summarized by the following topics:

- Reconstruction: Capturing and encoding three-dimensional data of time varying scenes, avoiding classical problems of mutual element occlusion and noise.
- Representation: Encoding time varying sequence of point clouds into a layered representation with optimized points of view for each layer
- Visualization: Development of point cloud visualization techniques and alternative visual representations
- Development of novel applications with the 3D Flashback framework, specifically in the context of dance performance and arts.

1.3. Contributions

Our main contribution is a novel image-based representation and encoding algorithm: the Multiview Layered Depth Image (MVLDI) , described in Section 3.2. Comparing to previous approaches, Multiview+Depth (MVD) and the Layered Depth Image (LDI), the MVLDI has better redundancy detection and a smaller number of generated layers. Also, it supports wide-baseline scenarios, which was only possible using MVD and no redundancy estimation. We achieved higher compression rates than the alternatives (MVD, LDI) without discarding necessary data. MVLDI provides a more efficient representation for a single frame and is applicable to video scenarios.

The viewpoint selection problem for MVLDI was also thoroughly explored, resulting in a detailed study of different alternative approaches (Section 3.3). Search-based

approaches were shown to be overall effective, however, using the average normal vectors of the biggest clusters allowed us to achieve better results in more complex scenarios. Also, we present a synthetic viewpoint generation algorithm for MVLDI, which can be applicable to image-based rendering (IBR) scenarios and complex point cloud visualization and representation.

Regarding other challenges in the 3D Flashback framework, our next big contribution was Stroke-based Splatting (Section 4.3.2), a novel rendering technique to visualize point clouds, that is easily applied to a free-camera navigation VBR application. We are able to offer equal or better visual quality than other rendering techniques (splatting, meshes), specially when dealing with noisy data or low-resolution clouds. The Householder formula was used to calculate tangential vectors, coherently orienting the surface aligned splats. When compared to the standard method used in splatting (covariance matrix), it is more resilient to noise, and creates a less variable tangential field, while still representing relevant details of the data.

We also introduced three different applications that use the 3D Flashback framework in the context of contemporary dance teaching and digital cultural heritage (Section 5). We presented a 2D-3D annotation system, and a Virtual reality approach, which allow choreographers and users to enhance recorded data of a dance performance with localized temporal annotations. Also, we presented a case study of a study made on the work of the portuguese choreographer João Fiadeiro, using a 3D video as the media used to register the concepts present in his work.

Other implementation questions to the 3D Flashback framework were also introduced and discussed, such as a segmentation and synchronization (Section 4.2), and a calibration toolkit (published in Sousa et al. (2017)).

1.4. Publications

This section only lists those publications directly related to my thesis topic.

1. Creepy Tracker Toolkit for Context-aware Interfaces. M Sousa, D Mendes, RK dos Anjos, D Medeiros, A Raposo, A Ferreira, J Pereira and J Jorge. ACM Interactive Surfaces and Spaces (ISS), 2017

2. Multiview Layered Depth Image. RK dos Anjos, JM Pereira, JA Gaspar, C Fernandes. Journal of WSCG 25(2), 115, 2017

3. Stroke-based splatting: an efficient multi-resolution point cloud visualization

technique. RK dos Anjos, CS Ribeiro, DS Lopes, JM Pereira. The Visual Computer, 1-15, 2017

4. Capturing and Documenting Creative Processes in Contemporary Dance. CS Ribeiro, RK dos Anjos, C Fernandes. Proceedings of the 4th International Conference on Movement Computing, 7, 2017

5. 3D Flashback: An Informative Application for Dance. RK dos Anjos, JM Pereira, C Fernandes. ERCIM News 108 (108), 45, 2017

6. 3d annotation in contemporary dance: enhancing the creation-tool video annotator. C Ribeiro, RK dos Anjos, C Fernandes, JM Pereira. Proceedings of the 3rd International Symposium on Movement and Computing, 41, 2016

7. Creepy Tracker Toolkit for Context-aware Interfaces. M Sousa, D Mendes, RK dos Anjos, D Medeiros, A Ferreira, A Raposo, JM Pereira, and J Jorge. In Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17). 2017.

8. Virtual Reality Annotator: A Tool to Annotate Dancers in a Virtual Environment. CS Ribeiro, RK Anjos, C Fernandes. Part of the Lecture Notes in Computer Science book series (LNCS, volume 10605)

1.5. Dissertation Outline

We will start with a general overview of video-based rendering, related techniques, and a classification of different lines of research based on the user interaction paradigm used by the applications (Chapter 2). A brief discussion is presented together with the classification, highlighting current problems that need to be solved in order to implement the 3D Flashback framework.

Following, we will describe our proposal for the main research question of this PhD Thesis, data representation (Chapter 3). Initial experiments are presented, followed by the description of the Multiview Layered Depth Image, and view generation algorithms. We also discuss the results of temporal compression in this data.

Next, a full description of the 3D Flashback framework is given (Chapter 4), approaching the reconstruction and visualization problems and sub-topics, and stroke-based splatting, the novel algorithm for point cloud visualization.

We then list three case-study scenarios in the form of applications developed in the

context of the BlackBox project (Chapter 5), and that are concrete examples of wide-baseline VBR applications which use our presented framework.

Lastly, a summarized list of conclusions is given, focusing on the contributions and limitations of the presented work, presenting clear guidelines for future work.

2
Related Work

2.1. Introduction

Video-based rendering is a topic that combines computer graphics and computer vision; competences from both areas of knowledge are needed. A great effort is made by each community to build the bridge between the two areas. Video-based rendering is without a doubt a challenging field of work.

Different paradigms of user interaction have been proposed for VBR applications, creating widely varying lines of work and methodologies to be followed. Each group of applications face different problems, and apply different methodologies and steps on each level of the typical VBR pipeline (Section 2.1.3). A classification scheme based on these internal decisions is very sparse and does not have clearly identifiable classes of techniques and methodologies.

This chapter reviews and classifies VBR works in different groups with the most high level classification parameter being the user interaction paradigm, while giving insight on the chosen methodologies, data representation, and techniques in the VBR pipeline. What are the lower level requirements for higher level techniques and applications? And more specifically for our research goal: How current data representation methods be expanded and developed to push applications into different

objectives?

This chapter will start by defining video-based rendering, the taxonomy to be used in this thesis, and the VBR pipeline. A review of image-processing techniques related to VBR will be presented, following by the state of the art report on Video-based rendering applications and data representation, comparing the most popular trends and grouping similar techniques in general categories. Finally, conclusions and insight will be given on what is the current trend of research, which guided the research goals for this thesis.

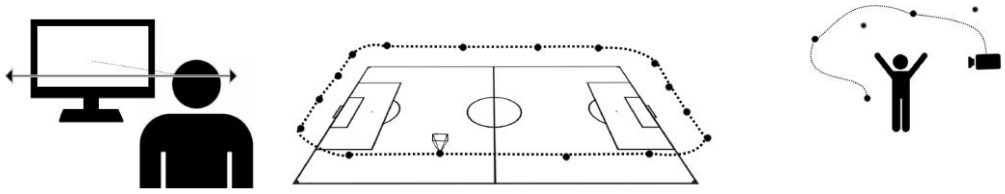
2.1.1. Video-based Rendering Definition

Video-based rendering is a term that has been applied to a wide range of techniques, sometimes in a more broad way than it usually is, and other times focused on only a specific type of application. So it is important to establish the definition that will be used on this chapter. The term was firstly used on the article bySchödl et al. (2000) referring to image-based rendering techniques extrapolated to the temporal domain, using two-dimensional images of a scene to generate a three-dimensional model and render novel views of the scene.

The book from Magnor (2005) defines video-based rendering as the process of fusing image-based rendering with motion capture in order to generate a novel view. Borgo et al. (2012) on their more broad survey classifies at a top level the techniques under the definition of video-based graphics (a more generalist definition for VBR), focused on creating new content (other videos or 3D reconstructions) based on video input, and video visualization that would encompass the attempts of allowing the user to see video from new/synthetic points of view not previously recorded.

The survey from Stoykova et al. (2007) focused only on 3D time-varying reconstruction, more in line with the classical definition of Magnor (2005), and would be only a subset of the previous classification, as also Szeliski (2005) who stays with the classical definition.

The common ground among all different definitions made at different points in time is the shared goal of creating novel viewpoints of a certain scene, not necessarily sharing a methodology as suggested by Magnor, or a specific type of input, as suggested by Borgo et al. We also consider scenarios where depth information or three-dimensional models are used combined with videos, since the goal of view synthesis is still shared. Considering this, we define VBR as **the process generating**



(a) Head-face parallax (b) Navigation through viewpoints (c) Free virtual camera

Figure 2.1: Different user interaction paradigms for VBR, which are the basis for our classification.

novel views of a recorded event on video.

2.1.2. User Interaction Based Classification

The chosen definition accommodates a large group of works which have considerable differences among them. Not only different devices are used for input, but also processing techniques, and type of data representation will differ considerably from one work to another. Due to this fact, defining clear groups of applications considering every applied technique is not viable. Few attempts of classifying VBR techniques as a whole have been made, with surveys commonly focusing on classifying each type of application or lower level techniques.

Authors have classified techniques according to taxonomies based on external aspects of the application such as level of automation, type of output and input information (Borgo et al., 2012), or had to focus on a more specific domain of applications where classification is simpler (Stoykova et al., 2007).

We found that the chosen user interaction paradigm for a VBR application is ultimately the deciding factor on three key aspects of a VBR technique: **View generation methodology, capture setup, and data representation.** Figure 2.1 shows the different paradigms found on the reviewed literature, which will be analyzed in depth in section 2.5.3. Figure 2.2 shows the choices for each one of these aspects according to the user interaction paradigm of the application. By classifying the techniques according to the five possible combinations of choices that can be made, we have clear different classes of works that one can easily identify and apply to different real world problems. Each one of the described aspects and grouping of applications will be described in Section 2.5.

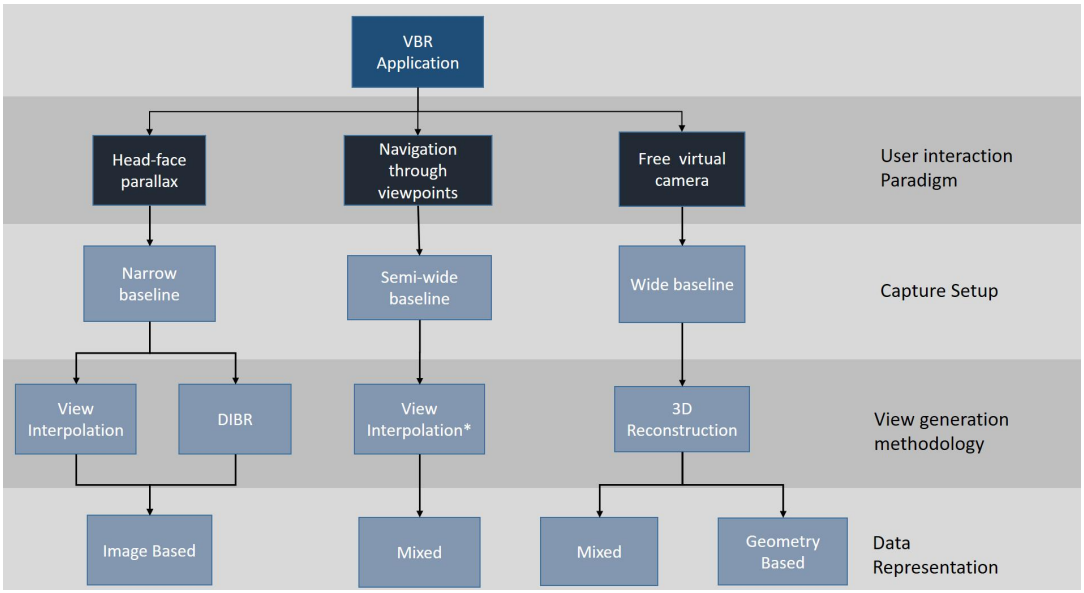


Figure 2.2: Diagram showing the used classification scheme for this survey. User interaction paradigm defines what capture setup is needed, which relates closely to view generation methodologies and data representations.

2.1.3. Video-based Rendering Pipeline

Although VBR techniques are highly varying on the required steps to achieve its objectives, one can assume a typical pipeline operations that are performed on a typical VBR process. Not all of the steps are essential depending on the application objective or input devices, and each one uses different techniques on the different steps. The different classes presented previously (Section 2.1.2) will typically use the same building blocks in their methodologies.

This VBR pipeline is implicit across all surveyed works, and also referred on the Survey from Borgo et al. (2012). Figure 2.3 shows all of these steps, including the more popular techniques in each step that were applied to the VBR works reviewed in this survey. This pipeline consists of the following steps:

Capture: acquiring the data that will serve as input for the process, which can be performed using different setups and devices.

Low-level processing: processing the raw input from each input source and adding low-level meta-information to these.

High-level processing: combining images and/or low level meta-information in order to achieve a higher goal that is not yet a VBR application on itself.

VBR Application: An application that can be used to generate novel views through

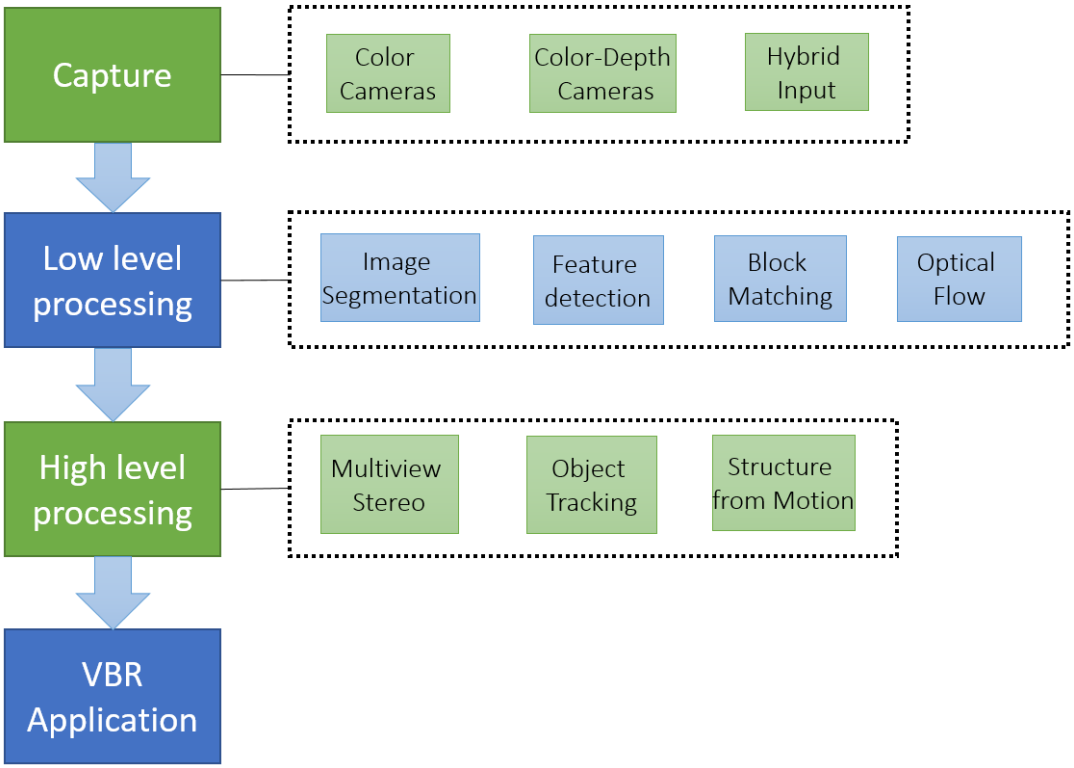


Figure 2.3: Diagram showing each level of the typical VBR pipeline, and the reviewed group of associated techniques in this survey. VBR Applications may use different methodologies and representations, but share the same goal.

different methodologies, and specialized data representations, and can be used by a common user.

Understanding of the techniques that are applied in a VBR process is essential to describe the available methodologies, and the existing challenges in each one of them. For this matter, Sections 2.2 to 2.4 will give insight on each step of this pipeline, reviewing and comparing techniques.

2.2. Data Capture

Capturing images to be used in a VBR system implies choosing the input devices and the physical setup in accordance with the task at hand. Input data including or not depth, field of view, scene occlusions, calibration and synchronization are examples of aspects to consider for video-based rendering applications.

Besides conventional color cameras, color-depth, laser scanners, and mixed inputs have been used on VBR and IBR applications.

2.2.1. Color Cameras

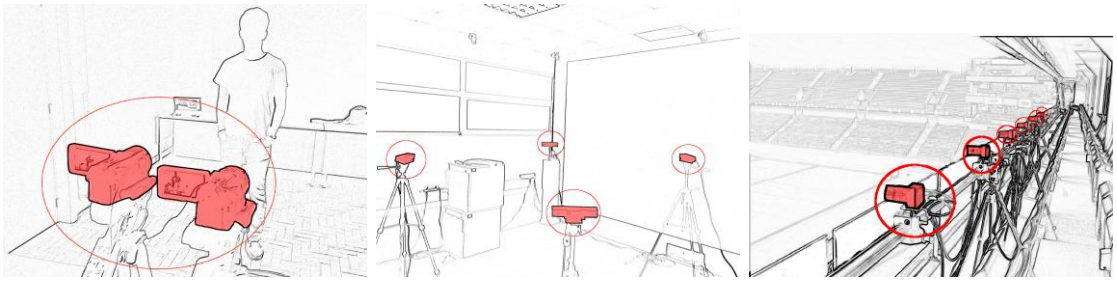
Main efforts in image and video-based applications are focused on capturing images with conventional color cameras (Goorts et al., 2013; Hauswiesner et al., 2011; Furukawa and Ponce, 2010; Carranza et al., 2003; Vogiatzis and Hernández, 2011), not only due to the lower cost of the devices, but the popularity of the developed methodologies (code publicly available) and the amount of data already available that could be used for applications such as showed on the work of Ballan et al. (2010). Besides being a bigger challenge than using more complex and informative data, it is of great interest to be able to use raw images for a VBR process. For information on what type of camera should be used for VBR, we refer to the book from Magnor (2005) which gives insight into this matter.

2.2.2. Color Depth Cameras

Another input device that has been recently popularized on VBR applications is the color depth camera. Asus Xtion Pro, Intel RealSense, and most popularly The Microsoft Kinect Sensor have been used due to their real time nature and low-cost. Depth sensors were already an option on the past (Curless and Levoy, 1996) but recently they were made more accessible and complete with other built-in functions. Differently from traditional laser scanners, these devices try to operate in real time, making them suited for VBR, unlike traditional scanners (Koutsoudis et al., 2014; Huang et al., 2013; Besl, 1988) which deliver high quality results, but have long capture times. Color-Depth cameras have been applied for stereo view generation (Arieli et al., 2012) using a single device as input. They are not set back by textureless regions as image-based stereo methods (Lee and Ho, 2011), but might suffer from interference from sunlight in outdoor scenarios.

2.2.3. Hybrid Input

Duan et al. (2012) showed that is possible to perform fusion between depth maps from stereo cameras and Kinect sensors in real time, having an overall better result than using a single device. The work from Goesele et al. (2006) is an example of another type mixed input that combines the raw images with an estimated bounding box for the object to be scanned. Also Ballan et al. (2010) take other information as input such as available 3D models for a prior reconstruction of the scenery and better



(a) Narrow setup with color cameras for stereo matching

(b) Wide baseline of a low-cost setup for a 360 degrees capture of a subject.

(c) Semi-Wide baseline setup for Sports

Figure 2.4: Different capturing setups for VBR with different input devices.

positioning of the cameras, since the input videos are not calibrated by default. The 3D model input does not always guarantee a better result, but having an initial geometry estimate does improve with the efficiency of the technique, as shown by the image-based rendering review from Shum and Kang (2000).

2.2.4. Baseline of the Data Acquisition Setup

Multi input setups are the typical scenario for VBR. Devices can be placed in a narrow, wide, or semi wide-baseline setup as seen on Figure 2.4. On the first, the cameras are placed closer to each other with little disparity between adjacent views, usually with each device parallel to each other. A wide setup typically aims to capture a scene or object from all different perspectives, having the cameras placed further away from each other, where disparity between views is now desired, not avoided. The semi-wide scenario would be a step in between where disparity is avoided but different viewpoints are desired. The relation between these decisions and the applications, and alternative setups will be further described in Section 2.5.3

On multi-streams approaches there is also the need of extrinsic calibration for the cameras, i.e. know the relative positions between them. In controlled environments this can be done by using markers detected by the camera (Sturm and Maybank, 1999; Carranza et al., 2003), but on dynamic environments the most common approach is to track features using structure from motion (Ballan et al., 2010; Izadi et al., 2011; Newcombe et al., 2015), providing a reliable position calibration for the camera. A parallel problem to this is the stream synchronization problem, which can be solved by an external centralized trigger on controlled scenarios (Carranza et al., 2003; Goorts et al., 2013). Audio stream aligned can be used on uncontrolled

scenarios (Duan et al., 2012; Ballan et al., 2010).

2.3. Low-level Processing

This section describes the techniques that work directly on an unprocessed image or video with the purpose of attaching meaning to the analyzed content, working at the lowest level possible. Techniques such as Object detection that might use multiple low-level techniques will be described in Section 2.4. In this section we will approach block matching, optical flow estimation, feature detection and image segmentation.

2.3.1. Image segmentation

The process of image segmentation; identifying and grouping pixels that share a certain meaning, is one with a generic purpose, that must be fine tuned and trained to solve each desired problem. Image segmentation plays a major role in several applications since usually the focus of the applications is to enhance the visualization of a single object that can be considered in the foreground (Ballan et al., 2010; Carranza et al., 2003; Liu et al., 2010), and also to extract silhouettes for reconstruction.

A general review on Image segmentation methods is given by Pal and Pal (1993), and specifically foreground detection by Bouwmans et al. (2010).

Supervised methods: These methods are used on controlled environments, or rely on some degree of user input. The simplest method for image segmentation is Chroma-key (Belmares-Sarabia and Chayka, 1989), where the background is already colored in a specific color before previously to the recording (Liu et al., 2010; Hauswiesner et al., 2011) , being easily identified in post processing. Ning et al. (2010) use user input in a more flexible scenario. On a first step a simple segmentation process such as mean-shift (Cheng, 1995) is used, then user input is requested to insert two distinct simple markers on a foreground object and on a background object. Regions containing each of the distinct markers are tagged as being foreground and background, and then a merging process is performed to achieve the real desired segmentation.

Unsupervised methods: We highlight three different unsupervised approaches found in the VBR related literature. Contour detection and merging (Arbelaez

et al., 2011; Del Bue and Agapito, 2006), which is efficient in scenarios with low frequency images. A probabilistic approach (Alpert et al., 2012), where where pixels are grouped bottom-up considering not only intensity values, but likelihood of having the same texture, appropriate for higher frequency images. And a higher level approach, which uses SIFT features matching and clustering to detect the same object in different situations (Joulin et al., 2010; Lowe, 2004)

Video techniques: Change detection, is the video-exclusive image segmentation approach. The simple yet popular approach of image differentiation (Rosin and Ioannidis, 2003) works by simply thresholding the difference between two images at a global level. There are also more complex techniques such as Change vector analysis (Bruzzone and Prieto, 2000) and Image ratioing (Di Stefano et al., 2003). Although popular, these techniques suffer with noise and illumination variation (Brutzer et al., 2011). A more detailed description and several references for each class can be seen on the article by Radke et al. (2005).

2.3.2. Keypoint extraction

Keypoint extraction is a classical low-level image processing challenge which consists on finding and describing interest points or regions on a certain image. Keypoint detectors usually look for patches with image points with high spatial directional derivatives i. e. the so called image corners. Classical approaches that are still applied are (Harris and Stephens, 1988) corner and edge detectors and Lindeberg (1998) difference of gaussians.

Global Keypoint analyze the image as a whole and provide us with information on a general level, such as illumination and color. Since they are unable to provide us with precise object information, their use in Video-based rendering is limited. Therefore we will be focusing our description on Local Keypoint descriptors. The main use of these techniques in video or image based rendering is to detect specific points of an object of interest, in order for it to be identified on a different image or on a future frame of a video sequence. A more detailed view on detectors and descriptors for image-based applications, refer to the survey from Moreels and Perona (2007) and the comparative work from Boyer et al. (2011).

Classical approaches: After detecting key features, descriptors are created to characterize the image around the location pointed by a detector. Scale Invariant Feature Transforms (SIFT) (Lowe, 2004), are the most popular ones, calculating the descriptor on different scales and rotations, being able to be matched against a

Figure 2.5: Output of a flow analysis algorithm Xu et al. (2012) Different colors show the amount of movement between frames.

large database of images of the same object in different scales and positions. Other popular approaches are shape descriptors (Belongie et al., 2002) and Speeded-Up Robust Features (SURF) (Bay et al., 2008), the latter being faster to calculate than SIFT. An extension for SIFT applied for face meshes resulting from face scans has been developed (Maes et al., 2010), showing how adaptive SIFT descriptors can be.

Video techniques: Specifically for videos, some approaches extend the SIFT idea to adding a temporal dimension to it. Scovanner et al. (2007) use it for action recognition, using a sequence of sift descriptors to represent each action. Ben Ahmad et al. (2011) accomplish a similar goal using an accordion representation, that tries to place pixels with high temporal correlation in adjacent spaces. Moving points are detected after background removal and motion detection, so it is better suited for single object tracking. The recent work from Tang et al. (2014) on Bold Features is also well suited for video scenarios where light variance and blurred images are a common issue. On these specific scenarios its performance was better than SIFT or SURF features.

2.3.3. Optical Flow estimation

Optical flow is a velocity field in the image which transforms one image into the next image in a sequence (Horn and Schunck, 1981) (Figure 2.5). It is mainly used by View interpolation in VBR techniques (Section 2.5.1.2). In this section, differential methods (local and global) are discussed. For a more detailed discussion on alternative approaches, please refer to the survey from Barron et al. (1994) or the more recent technique comparison by Del Bue and Agapito (2006) or Philip et al. (2014) .

Differential methods are the most common techniques to do flow estimation. Assuming only small changes between consecutive frames, spatio-temporal derivatives of the image intensity or other measures are calculated between frames, estimating the optical flow as velocity vectors. Differential methods can be roughly classified as local or global.

Global methods: these rely on a smoothness operator or a regularization factor applied to the whole image. They do not suffer from the aperture problem (Bertero et al., 1988) of local methods, where motion information can not be estimated since the indicators of motion of the analyzed object are outside the area looked upon. This problem is normally noticed on large motions that are covered in the whole picture. The downside of these operators is that since they are applied to the image as a whole, they are more sensitive to noise, propagating errors through the whole image (Bruhn et al., 2005). The most referred global method is the work by Horn and Schunck (1981) , where the authors claim the apparent velocity of the brightness pattern varies smoothly through the image.

Local methods: while suffering from the aperture problem, they avoid the error propagation of global methods as they only calculate the motion flow using local information from the pixel neighborhood of the patch currently being evaluated. The most popular local method is the work from Lucas et al. (1981), where the spatial intensity gradient of the images is used to find a match in the following frames using Newton Raphson iteration. The result is a more robust but less dense flow field. More recent implementations such as the work from Bruhn et al. (2005) attempt to use the best of both algorithms. By cumulatively adding the local contributions to the histogram, the regions with lowest contribution are detected as the ones with highest confidence. This measure is then applied to the global smoothing function, that will have a higher effect on these areas.

2.3.4. Block Matching

Techniques in this category are one of the classical ways of extracting information from moving pictures. These algorithms consist in dividing a frame into blocks of size N x N, and matching them against equal sized candidate blocks on the next frame on an area around the original location. After finding the best matched block, the difference (Motion Vector) between time t and t+1 is recorded. It is mostly used in video compression to remove temporal redundancy within frames, detecting what are the actual differences that need to be transmitted in order to fully reproduce

the video.

The reference algorithm for the implementation of MPEG4 (Zhu and Ma, 2000) applies a search strategy called Diamond search that restricts the search space using a diamond shaped pattern. The results do not equal a full-search, but greatly outperform its prior strategies such as cross search (Ghanbari, 1990) and three-step search (Koga, 1981). A more recent work (Cuevas et al., 2013) poses the problem as an optimization task for the Motion Vector and uses an Artificial Bee Colony (ABC) algorithm to achieve a fast answer without compromising the result. They claim their approach does not easily get trapped in local minima due to the non-linear characteristic of the ABC operators, and that the efficiency is comparable to the fastest algorithms.

The surveys from Huang et al. (2006) and Yaakob et al. (2013) describe in more detail the different approaches and provide in depth comparison between them.

2.4. High-level Processing

After obtaining features, image segments, or optical flow, this information can be used as input for a new processing step that outputs even higher level of information about the capture, without creating a user-targeted application. These are usually referred as High-level techniques.

Two popular straightforward techniques are View interpolation (Szeliski, 2005) and Visual Hull Estimation (Matusik et al., 2000). The first will be described in Section 2.5.1.2. The second one combines silhouettes from different viewpoints to create an estimate of a 3D shape, and will be discussed in Section 2.5.1.1.

In this section we will review multiview stereo (MVS), Photo-consistency reconstruction, Object tracking, and structure from motion (SfM) in further detail due to their widespread use, and existing variations on the specific methodologies.

2.4.1. Multiview stereo and Photo-consistency reconstruction

Multiview stereo (MVS) has been the most standard approach for 3D reconstruction, seeking to combine images captured from known positions to reconstruct a complete

3D model of a given scene (Okutomi and Kanade, 1993). The work from Furukawa and Ponce (2010) is a recent example of classical MVS which introduces a novel patch-based representation to support the reconstruction task.

Photo-consistency based reconstruction is an alternative approach that can be applied when a rough estimate of the scene is provided. The recent work from Lafarge et al. (2013) proposes an iterative improvement approach: the starting point is a rough estimate of the 3D model that one wants to reconstruct and, interactively, it is segmented and the extracted primitives are back-projected into the input images to apply photo-consistency measurements. The process is repeated until no more improvements can be made and the back projection closely resembles the input images. Mixed approaches (Vu et al., 2012) use MVS or a visual-hull algorithms(Esteban and Schmitt, 2004; Kolmogorov and Zabih, 2002; Fan and Ferrie, 2010; Kopf et al., 2013) to estimate a model and then apply photo-consistency measurements.

MVS has also been used in video scenarios on video sequences (Liu et al., 2010; Vu et al., 2012). These works strongly state that classical MVS can be efficiently applied to videos and deliver good results. For a more detailed insight and classification on MVS , we refer the survey from Seitz et al. (2006)

2.4.2. Structure from Motion

Structure from motion (SfM) consists of the extraction of a 3D structure from a set of 2D sequence images based on the motion of the structure. It normally works under the assumption of having a static environment, so the motion is considered as a result of camera movements. An example can be seen on Figure 2.6. The work from Chen and Pinz (2004) performs the classical SfM approach of feature detection and position tracking, being able to recover a sparse 3D structure from an image sequence after estimating the camera parameters. SfM is a less costly algorithm than MVS and it can be used to extract a rough estimate of scene structure before a more accurate reconstruction step Agarwal et al. (2011). Recent work by Crocco et al. (2016) uses object detection and tracking to perform SfM instead of feature matching, producing better results in scenarios with several objects.

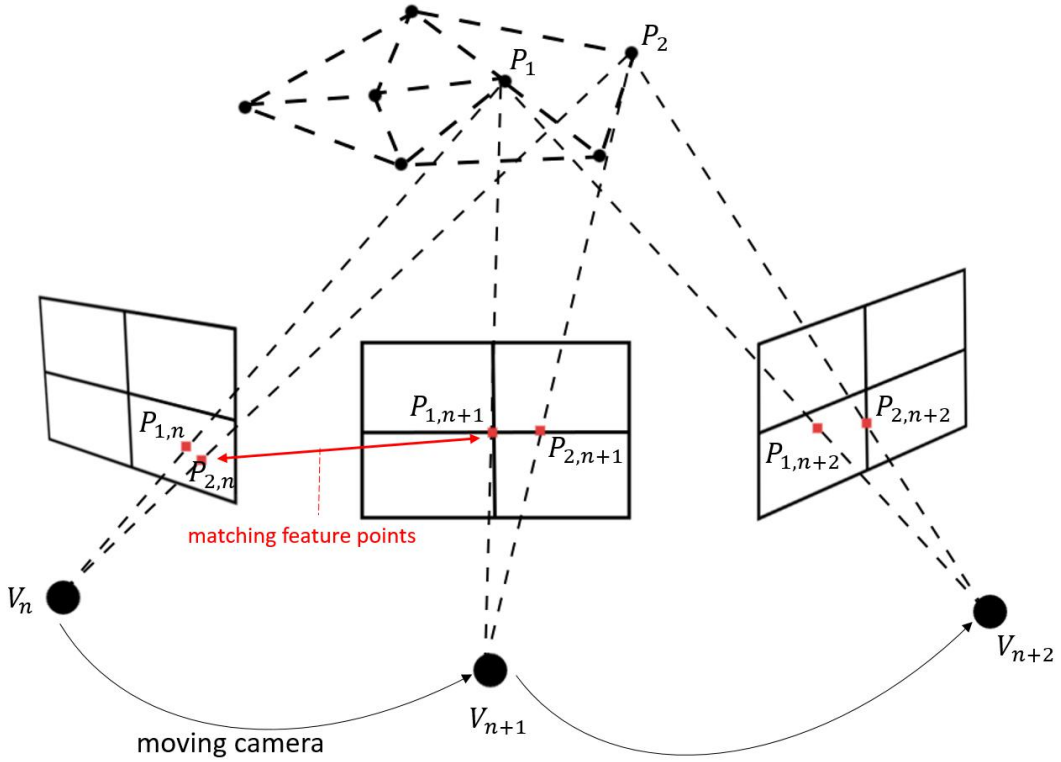


Figure 2.6: Structure from motion. Matches are found along different time stamps in a moving camera in order to estimate the structure of an object.

2.4.3. Object Recognition and Tracking

Object recognition and tracking are methodological components found in many computer vision and VBR applications. Usually these are real-time solutions (Henriques et al., 2015). The common approach to perform object detection is based on using a set of features to identify the object in different scenarios which will be used to train a classifier such as a decision tree, neural network or support vector machines.

Human Skeleton recognition and tracking has been applied on VBR applications where the target is a human performer (Gall et al., 2009; Germann et al., 2010). By tracking subjects positions and bones, specialized data representation can be used, as will be discussed on section 2.5.2.

Nevertheless challenges remain on untrained scenarios. Work from Cabral et al. (2015) presents a solution for weakly supervised scenarios, where the labeling process for training is replaced by semantically tagging the images. The surveys from Yilmaz et al. (2006) and Prasad (2012) give an insightful look on these techniques and classifying them accordingly.

2.5. Video-based rendering applications

As stated in Section 2.1.1, the main objective of VBR applications is the generation of novel views. We selected thirty two articles from over the last 15 years which share this objective, yet use different approaches. We started by selecting papers mentioned in previous surveys, followed by other works in the venues these were published, and finally more recent work from such authors. We sought to answer a group of questions for each one of them:

1. What capture device was used?
2. Which lower level techniques were applied?
3. Which higher level techniques were applied?
4. What view generation methodology was used?
5. What was the data representation used for that application?
6. What was the capture setup used?
7. What is the user interaction paradigm applied to this ?

Questions 1-3 give us insight on individual decisions each one of the works make, but did not reveal clear groups of applications, or informed us about high level methodologies. This is due to the fact that these decisions are relatively low level, and techniques are applied with different purposes and in different combinations, not necessarily defining an approach or application.

Questions 4-6 are higher level decisions which clearly relate to each other and allow us to classify different works into categories. Methodologies for view generation (4) were identified in our review, which have strong relations to other of the raised questions, and also allow different types of application for each one of them. Data representation (5) will decide what data is stored, and what can be generated in these novel views. Finally, the capture setup (6) is directly related to the user interaction paradigm (7) of the application, since it decides the spatial limits of the interaction. We considered these four aspects to be the most relevant on defining a VBR application. Nevertheless the user interaction paradigm (7) was found to be the key deciding factor on what approach is used, as discussed in section 2.5.3.

The following sections (2.5.1 2.5.2 and 2.5.3) will describe the answers to the last four questions listed before, giving insight on each one of the reviewed works, explaining its relevancy in a VBR application and the user interaction paradigm in

hand (Section 2.5.3). Finally a summary of these answers along with a taxonomy (2.5.4) will be presented.

2.5.1. Novel view generation method

Having captured an event from multiple viewpoints, unrecorded visualizations can be generated through different processes. The chosen methodology will depend on the available data (3D information, images, depth values, etc) and the desired interaction (navigate freely vs. recorded viewpoints).

Older definitions of VBR mentioned on Section 2.1.1 defined VBR through the used methodology. Schödl et al. (2000) and Magnor (2005) defined it as processes that necessarily required reconstruction. The fact that the field evolved in different directions and newer processes and applications were created, we decided to use a definition based on goal only, and use the methodology as one of the classification parameters of a certain work.

2.5.1.1. 3D Reconstruction and rendering

The classical definition of VBR was grounded on 3D reconstruction and rendering procedures to generate views (Schödl et al., 2000) since this resembled the traditional process to generate novel views in computer graphics. Rendering 3D models into 2D photo-realistic images accordingly to the position and orientation of a virtual camera is a straightforward task that has been well documented and investigated by the community. When 3D information about the scene is available, any desired viewpoint can be rendered through this process. The outline of this process can be seen in Figure 2.7

In the VBR context, the 3D Reconstruction step poses a challenge because the initial input of the process does not commonly provide three-dimensional information. The inclusion of the recent depth sensors in the capture process could fix the problem but as mentioned in Section 2.2, using such sensors is not always viable, so we must still consider 3D reconstruction without direct 3D information from the input video streams.

As we are going to see next, despite of different approaches to provide 3D information for performing the 3D reconstruction, the novel view creation is accomplished by

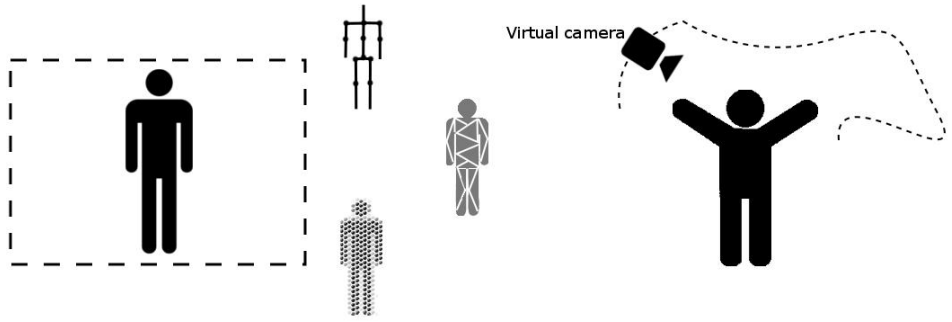


Figure 2.7: Outline of the 3D Reconstruction and rendering view generation method. Captured data is used to create different types of representations (3D Reconstruction), which are then used differently to create a 3D visualization (rendering).

executing afterwards the classical rendering process with the available 3D models or structures that were estimated from the input.

When the focus of the application are human performers (e.g. sports and dance applications), very simplistic 3D information such as an estimated skeleton can be sufficient for novel view generation. Players are segmented from the background, and their skeletons are recognized from the poses captured in video. On the works of Gall et al. (2009) and Li et al. (2013), a mesh is estimated using a visual hull for the performer so it can be applied to the tracked skeleton. Stoll et al. (2010) and Wu et al. (2013) move this task to a pre-processing step where depth sensors are used to create an animated model of the performer. One mentioned drawback is that changes in the outfit or hair of the performer are not be supported.

Germann et al. (2010) has a similar but unique approach, where the same process for estimating the skeleton is used, but instead of applying a 3D mesh to it, segmented billboards of each body part of the performer are applied to the tracked skeleton, this approach is not a pure 3D reconstruction case since the applied textures are view interpolated. We chose to describe it here due to the similarities to the previous approaches.

Volino and Hilton (2013) and Imber et al. (2013) use a initial capture of the performer to construct a texture map, which will be applied to the estimated visual hulls in each frame. A skeleton is not estimated on these works, instead a sequence of visual hulls is calculated.

Finally, the most straightforward approach to 3D Reconstruction relies on directly estimating depth information from camera inputs, or depth sensors, creating complex three-dimensional structures that will be used for rendering. Zeng et al. (2013) and Kuster et al. (2014) use directly the input from the Microsoft Kinect for that

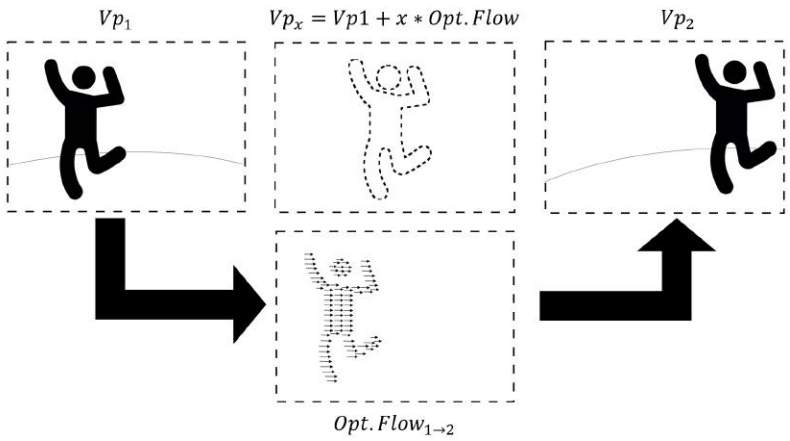


Figure 2.8: Outline of the View interpolation method. Optical flow between adjacent viewpoints is estimated, and interpolation is performed to create an intermediate point of view.

task. Google Tango Google (2014) and the work from Liu et al. (2010) used multi-view stereo to estimate depth information, and on the latter, a visual hull is used to define the limits of the human performer that is being captured, refining the MVS process.

2.5.1.2. View interpolation

When the required novel views are close to a previously recorded video stream, 3D reconstruction step may not be necessary to perform the rendering operation. Chen and Williams (1993) described this process on their pioneer work. This approach introduced in 1993 allowed very complex scenes to be rendered through this process, since it is not reliant on the complexity of the objects to be rendered. Szeliski on his survey (Szeliski, 2005) presents this methodology also present in his own research as one of the basic building blocks for VBR applications.

The scene is captured with an array of aligned cameras, and the relative position between pixels from different viewpoints is estimated through the optical flow from one point to another. These vectors are stored in a "morph map", a disparity matrix, which will be used to interpolate the values between each one of the viewpoints and generate the new images on the unrecorded viewpoints, as seen on Figure 2.8. If the changes are parallel to the viewing plane, the interpolated result is perfect. Also, as mentioned before, the closest the images are to the original viewpoints, the better the estimated results.

One relevant reference is the work from Kanade (Kanade, 2001) about the coverage

of the Super Bowl XXXV, where the broadcasting team, instead of individual users, was able to cycle seamlessly through the several cameras set up in the stadium to give more insightful replays. View interpolation and a rough reconstruction which is possible due to the playing field being known, are used to create transition frames between cameras. A similar recent product by Vizrt (Libero, 2014) has been extending the functionality to allow not only transition between cameras but also to generate other points of view.

Goorts et al. (2013) uses a similar methodology, but uses MVS to estimate depths for each point, and render better interpolated images. Similarly, Taguchi et al. (2008), Wang et al. (2014a), and specially Tanimoto (2012) have used MVS, but in order to represent the scene in the Ray-space using the plenoptic function.This representation allows an easier generation of views given the accurate estimation of this space. Tanimoto (2012) introduced specified devices to quickly create such representation for small scale objects. Ng et al. (2010) uses the same methodology but with a more object focused approach, improving the results in object boundary regions.

One interesting view interpolation work that must be mentioned is the one from Ballan et al. (2010) which applies this methodology for a different purpose: to navigate between casual uncalibrated captures of the same performance. A rough three-dimensional reconstruction of the background is performed using SfM to estimate each camera position. Then view interpolation is used to create transition frames between one viewpoint to the other. The performer is represented as a billboard naturally changes during transitions, and the background information is interpolated between viewpoints. This work extends the work from Kanade (Kanade, 2001) and Libero (2014) to a more casual scenario, where the capture is performed in an uncontrolled scenario.

2.5.1.3. Depth image-based rendering

This methodology has been acquiring popularity in the recent years since depth data is easier to be captured or estimated with modern cameras or specified sensors. Novel views are rendered through warping the Color Depth data into three-dimensional information, which then can be viewed through chosen viewpoints. This process can be seen in Figure 2.9. The work from Zitnick et al. (2004) can be considered one of the precursors of this line of research. In this work depth is estimated through MVS and used for depth image-based rendering (DIBR). The resulting dancers

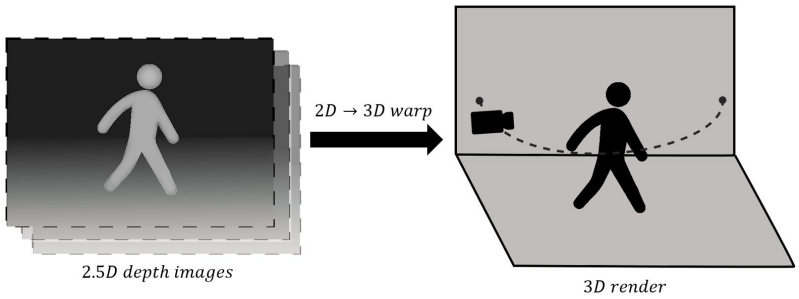


Figure 2.9: Depth image-based rendering. A set of 2.5D depth images is warped to create a 3D render that can be visualized from a set of positions.

data set has been used as a standard benchmark in the majority of work described below.

The novel view generation methodology is the same but each group of works has focused on different aspects of the process. Yoon et al. (2005) and Merkle et al. (2007) have presented specific data representation for this field (2.5.2), focusing on compression of data. This line has been followed by several authors (Daribo and Saito, 2011; Yoon et al., 2007; Kirshanthan et al., 2014; Kim et al., 2015; Merkle et al., 2016).

Due to the fact that the estimated depth values might not create a complete scene due to occlusions, or depth discontinuities might exist due to differences in estimation from one viewpoint to the other, other works have focused on in-painting and hole filling. Daribo and Saito (2011) and Yang et al. (2011) techniques have worked towards this goal using different data representations.

2.5.2. Data representation

After going through the lower steps of the VBR pipeline, information about the captured scene is encoded in a suitable format for the chosen rendering process. We found three types of representation in the surveyed works. Geometry based representations, where the scene was modeled as a group of three-dimensional objects along the time. Mixed representations, where part of the scene is modeled through images, and part through geometry. And image based representations, where the scene is stored in bi-dimensional matrices with color and optionally depth.

Although it might usually be considered just an implementation detail, the data representation on a VBR process is tightly related to the chosen methodology for novel view generation, and also to the desired type of application. Different repre-

sentations enable the development of alternative methodologies for view generation. As seen on Section 2.5.1, reconstruction was performed in different ways, all creating different types of data.

2.5.2.1. Geometry-based

The most straightforward way to represent a scene is through geometric primitives. It has been the go-to approach in most rendering scenarios. On VBR, they result from a 3D reconstruction process, and employed in traditional rendering to generate novel views.

Animated meshes were used in several works (Gall et al., 2009; Stoll et al., 2010; Li et al., 2013; Wu et al., 2013) where the target of the visualization is one or more human performers which can be segmented properly in order to estimate the skeletons. Scenarios with large groups, occlusions, and close interactions pose challenging issues. When a static mesh has been captured in a previous step for that single performer (Stoll et al., 2010), this representation is very efficient.

When a skeleton can not be reliably tracked, Surfaces (Liu et al., 2010; Kuster et al., 2014), point clouds (Google, 2014) and Octrees (Zeng et al., 2013) can be used. These are classically used for static reconstructions, but can be applied in dynamic scenarios. Although more flexible and being complete representations (contain full and precise information about the objects in the scene), they are less efficient for VBR. Applying temporal compression requires specialized algorithms (Slomp et al., 2014), while image-based representations can apply video compression, which is always evolving. Geometry-based representations are usually applied in real-time applications where storage and compression is not an issue.

2.5.2.2. Image-based

Image-based representations are independent of scene complexity, being well suited for these scenarios. On the other hand, they are typically discrete, and do not allow certain rendering effects that require precise geometric information. Specifically for VBR, they have the advantage of enabling ordinary video compression techniques to be applied to them, which is not possible with geometry-based representations.

On several View interpolation scenarios (Kanade, 2001; Szeliski, 2005; Taguchi et al., 2008), ordinary video streams for each recorded viewpoint are the only information

about the scene in hand. Although effective, further work has shown that depth information is important not only for view-interpolation and DIBR when pursuing accurate results. Color plus depth video streams have been used for this matter Zitnick et al. (2004); Yang et al. (2011), where depth information is estimated through MVS or captured with specialized sensors.

Two other image-based representations have been presented as alternatives to RGBD streams. Multiview+Depth (MVD) by Merkle et al. (2007), and the Layered Depth Video (LDV) by Yoon et al. (2005). They couple color information with depth at the moment of encoding, and have been widely applied in VBR, with LDV being a more compact alternative, but with a lengthier encoding. We will get into more detail on these representations and their limitations in Section 3.2.1, since they are closely related with our main research topic.

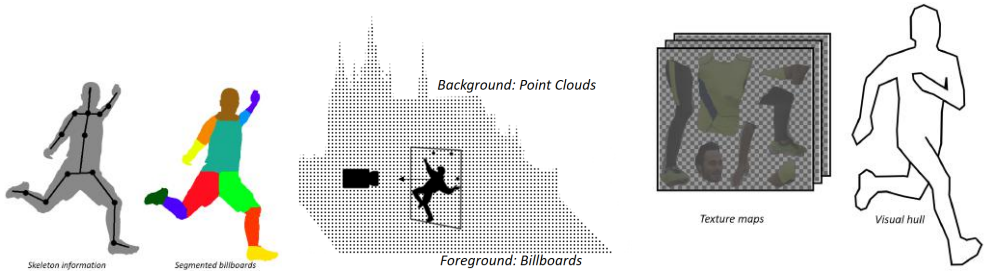
Finally, Plenoptic Videos (Tanimoto, 2012; Wang et al., 2014a) have been successfully employed on view-interpolation. They capture color and depth information from different viewpoints and represent it as the Plenoptic function (7D) (McMillan and Bishop, 1995), or the Lumigraph (Gortler et al., 1996), its 4D simplification. With θ and ϕ being the azimuth and elevation angle of the rays, and λ the wavelength, it is calculated at a position (V_x, V_y, V_z) in space, and on the VBR scenario, the function is 7D due to the time component. So we have the following form to the function, which can be considered a complete scene description:

$$p = P(\theta, \phi, \lambda, V_x, V_y, V_z, t) \tag{2.1}$$

Although it is a complete scene description, on a real scenario we cannot capture the scene from every possible viewpoint. In practice, data is captured with a narrow grid with several cameras, or cameras based on arrays of micro-lenses. This representation is used by sampling this function at the eye positions (v_x, V_y, V_z) representing the capture viewpoints, and interpolating the values given by each one of them to generate intermediate views. Such representation is promising for 3D television, but is still far from being accessible for research.

2.5.2.3. Mixed

Although MVD and LDI contain geometric information in the form of depth values, we still consider them as image-based representations due to the fact that they are stored as images, and warping needs to be performed during rendering to obtain



(a) Articulated billboards (b) Proxy geometry + Billboard (c) Visual hull + Texture-maps

Figure 2.10: Mixed representations with part represented by a geometric reconstruction, and part by sequences of images.

the three-dimensional values. Examples in this category are partly represented by sequences of images, and partly by geometry.

As mentioned in section 2.5.1.1 Germann et al. (2010) uses articulated billboards (Figure 2.10a) . Skeleton information (geometry) is stored alongside images which are interpolated and applied to each skeleton. Also the approaches from Volino and Hilton (2013) and Imber et al. (2013), which use a simplified mesh through a visual hull (geometry) combined with sequences of textures (images) that are mapped into it (Figure 2.10c). Ballan et al. (2010) has a similar approach but keeping the background geometry static since it is only used to track positions of each viewpoint in order to generate the transitions (Figure 2.10b).

Finally Ng et al. (2010) use the Plenoptic function representation, but segmented to individual objects in the scene, which can be considered a mixed representation, due to the fact that individual objects in the scene are separated from each other, making the representation more tied to the content of the scene than other image-based representations.

All of these representations aim to combine advantages from both worlds. Having a three-dimensional representation allows one to generate novel viewpoints further away from the original recording points, and using image-based representations, data compression is considerably easier to be applied, and the representation complexity is scene independent. It is important to notice though, that in all of the reviewed works, strong assumptions about the storing content needed to be made. Typically mixed representations were used to represent human performers in controlled conditions, such as a studio capture setup, or a sports event where the layout and the captured elements are known.

2.5.3. Capture Setup and User interaction paradigm classification

Although the general goal of VBR is the same across applications, each one of them have different specific goals depending on the type of user interaction desired, as seen on Figure 2.1. On all reviewed works, we found that the interaction paradigm is tightly connected to the capture setup. According to the objective of the application, the setup will be adapted, and all other factors mentioned previously are then a consequence of this decision. Due to this fact, this section groups each work by the camera setup, and explain the typical application for each setup, and how it relates to the previously raised questions.

2.5.3.1. Narrow baseline applications: Head-face parallax

One interaction paradigm associated to a free viewpoint videos consists of a moving user in front of a screen while having the perception of depth through parallax. By adjusting the viewpoint to the position of the user's eyes, this effect is possible. Since the user performs movements in a parallel plane to the captured scene, novel views only need to be generated in this domain. For this purpose, a narrow capture setup parallel to the captured scenario will suffice for the desired results. Figure 2.11 summarizes this application group.

When a narrow capture setup is used, cameras and/or depth sensors are arranged in a line (Zitnick et al., 2004) or in a grid (Taguchi et al., 2008), according to the freedom of choice of views provided by the application. This setup is ideal for a performance type of recording, where the audience is supposed to be facing a stage from a certain direction.

Methodologies such as view interpolation (VI) and DIBR have good performance in this scenario due to the small disparity between adjacent viewpoints. 3D reconstruction will create incomplete results, since only one side of the object is being captured. VI has been used when depth estimation is not reliable enough for rendering, but used sometimes as an aid to the interpolation process. DIBR have been used in all other works reviewed in this chapter.

All strategies for this setup have used image-based representations because they are meant to work on any kind of data with no expected restrictions, and as mentioned previously, image-based representations are independent of the complexity of the

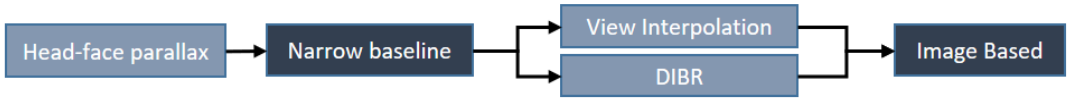


Figure 2.11: Head-face parallax application classes

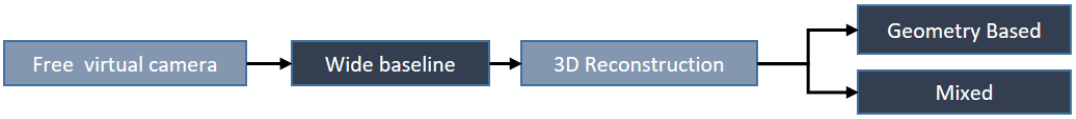


Figure 2.12: Free-camera navigation applications

scene.

2.5.3.2. Wide baseline applications: Free virtual camera

When the created application aims to generate novel views all around the subject of visualization, and not only on a parallel plane in front of it, a wide setup must be used. Interaction with the video is usually done indirectly, moving a virtual camera freely around the point of interest.

This type of setup has been used on scenarios where the focus of the video is a human performer in a controlled environment (Gall et al., 2009; Stoll et al., 2010; Li et al., 2013). A wide-baseline setup can be comparable to a single depth sensor moving widely around a scene for static reconstruction purposes (Liu et al., 2010; Google, 2014), since the camera will end up assuming positions equivalent to a wide-baseline setup.

Because the viewpoint disparity is too high for view interpolation and DIBR, 3D reconstruction was the methodology applied in all of the surveyed works. Regarding data representation, when stronger assumptions about the content of the scenes could be made such as in sports scenarios, or controlled environments, mixed representations could be used (Germann et al., 2010; Volino and Hilton, 2013; Imber et al., 2013). All the remaining papers in this category used different Geometry-based representations. Figure 2.12 shows the different choices that can be made in this application group.

Figure 2.13: Navigation through viewpoints applications

2.5.3.3. Semi-Wide baseline applications: Navigation through viewpoints

A small subset of works reviewed in this chapter aims a similar experience to wider setups, where the user can navigate in a full circle around a scene, but the content of the visualization is more complex than having a single performer. Similarly to Wide setups with mixed representations, strong assumptions can be made about the content, but the type of result desired is closer to narrow baseline applications. Either navigating through camera viewpoints, or generating intermediate viewpoints but not widely far from the defined grid of visualization. For this sense, a "less narrow", or "semi-wide" setup is used (Figure 2.13).

Instead of performing 3D reconstruction with view interpolation in some components such as the work from Volino et. al with articulated billboards (Volino and Hilton, 2013), the preferred approach is view interpolation supported by three-dimensional information about the scene. On sports scenarios (Kanade, 2001; Ballan et al., 2010; Libero, 2014), this information has been used to generate transition frames between viewpoints. Given the fact that the reconstruction is rough, the user never gets to properly visualize intermediate frames. The remaining works in this category (Ng et al., 2010; Goorts et al., 2013) create intermediate viewpoints, but use background geometry information to support this view generation process.

2.5.4. Summary and Comparison

As explained in Section 2.1.2 and seen on Figure 2.2 the different works can be separated in a hierarchy according to the aspects reviewed above. Each user interaction paradigm is closely tied to a camera setup, and to one or two methodologies or data representations. These two aspects are chosen according to the type of data to be captured.

Summarizing the reviewed aspects, DIBR and VI have been used in uncontrolled scenarios, where image-based representations can be applied. When strong assumptions can be made about the scene in hand, mixed representations have been used for

view interpolation or 3D reconstruction. Geometry based representations have been applied on generic scenarios with low requirements regarding quantity of data, or when the subject of the free viewpoint video was a human performer in a controlled environment.

Each application type will apply different techniques on the VBR pipeline introduced in section 2.1.3. The most common associations found between VBR related techniques and the presented application groups are:

Head-face parallax: feature detection has been used for establishing matches across views with low disparity, block matching for compression, and optical flow for view interpolation. On high level processing, MVS for depth estimation, and view interpolation.

Free virtual camera: Segmentation was the most important low-level technique applied in these works, due to the common focus on a single performer. Object tracking and MVS have been used on the high level processing step.

Navigation through viewpoints: Feature detection and segmentation at a low-level, MVS and SfM at a high level. SfM is relevant here due to the sparse reconstructions used on these techniques.

2.6. Discussion and Current Research Problems

The presented classification for VBR groups different approaches not only into clearly identifiable classes that share methodologies and problems, but also gives meaningful insight on how they operate on the traditional VBR pipeline. Figure 2.14 organizes the reviewed classes in a straight line according to similarity between each approach.

Table 2.1 list relevant surveys about every approached topic. We recommend those works as they provide more detailed investigation on the several covered topics by our paper, including previous VBR reviews for a more theoretical groundwork on the field Magnor (2005) or a different take on the same process Stoykova et al. (2007).

With our user interaction paradigm-based taxonomy, three different classes which have their own line of research were identified. Despite of the fact that they share

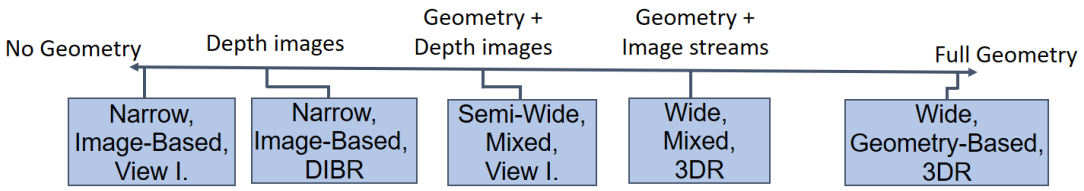


Figure 2.14: Classes of applications (setup, representation, view generation methodology) placed on a straight line according to the similarities between their approaches regarding to geometry used in their data representation.

similar techniques, each one aims to solve different application requirements.

We have noticed that geometrical information, including depth values, plays an increasingly important role in the three classes. This is justified by the hardware advances, namely, more powerful graphic cards and low-cost depth sensors availability. With new ways of interacting with 3D content arising such as Mixed and Virtual reality and the ubiquity of computers and interactive systems in our daily lives, the free virtual camera paradigm also increases in relevance. Wider baselines have been used due to the combination of these two factors. However, current data representations presented for these scenarios either need to make strong assumptions about the content (mixed), or are not easily compressed in the temporal domain (geometry).

DIBR has been a good example of an approach that integrates well the geometric component, being able to apply image-based representations which can be easily compressed in the temporal domain for transmission. However, it has only been applied to narrow-baseline scenarios due to the fact that the proposed image-based representations (LDI and MVD) face problems when dealing with wider baselines (Section 3.2.1).

Considering the current trends, we believe the main research goal in this area is **to develop an image-based representation to be used in the 3D reconstruction free camera navigation scenario.** With the continuously increasing requirements regarding viewing resolution, complexity of content and capture baseline, time compression aspects will become more significant.

Table 2.1: List of relevant surveys about the covered topics in this chapter

Survey topic	**References**
Block matching	Huang et al. (2006)Yaakob et al. (2013)
Change detection	Radke et al. (2005)
Segmentation	Brutzer et al. (2011)Bouwmans et al. (2010)Pal and Pal (1993)
Optical flow	Philip et al. (2014)Baker et al. (2011)
Features	Boyer et al. (2011)
Multiview stereo	Seitz et al. (2006)
Object detection/tracking	Prasad (2012)Swaminathan et al. (2014)Yilmaz et al. (2006)
Image-based rendering	Shum and Kang (2000)
Free viewpoint TV	Zhang (2007)
Representation	Kobbelt and Botsch (2004)Zhang and Chen (2004) Smolic et al. (2009)
Video-based rendering	Borgo et al. (2012)Stoykova et al. (2007)Szeliski (2005)Magnor (2005)

3 Data Representation

As briefly introduced in Chapter 1, the goal of this PhD thesis is to develop techniques which allow users to have a navigation experience of a 3D reconstruction of a video-captured reality from any point of view. A simplified overview of our approach can be seen in Figure 3.1. The first step (reconstruction) represents the capture process and low level operations performed to the stream (filtering, segmenting, etc). The second step (representation) is where the raw data is transformed into a more compact data format. The last step (visualization) contains decompression, rendering algorithms, and integration with applications.

This chapter describes the second step; representation. As introduced in Section 1.2, it is the central research question of this thesis. Image-based representations for rendering were initially introduced as more efficient alternatives to render geometrically complex scenarios. But as mentioned in Section 2.5.2, they have been

Figure 3.1: Simplified diagram of the 3D Flashback framework, explained in Section 4.1

tightly connected to video-based rendering (VBR) due to their temporal compression advantages. In the proposed 3D Flashback framework the data representation problem is undoubtedly central in enabling the desired type of applications.

The following sections will start by describing the initial experiments with a point-based representation that were performed to locate the main problems with this data format, and quantify the advantage obtained with the proposed solution. Following, our proposed layered representation, and an evaluation of different view generation algorithms. Lastly, we evaluate the effect of temporal compression in this type of data comparing to our initial approach.

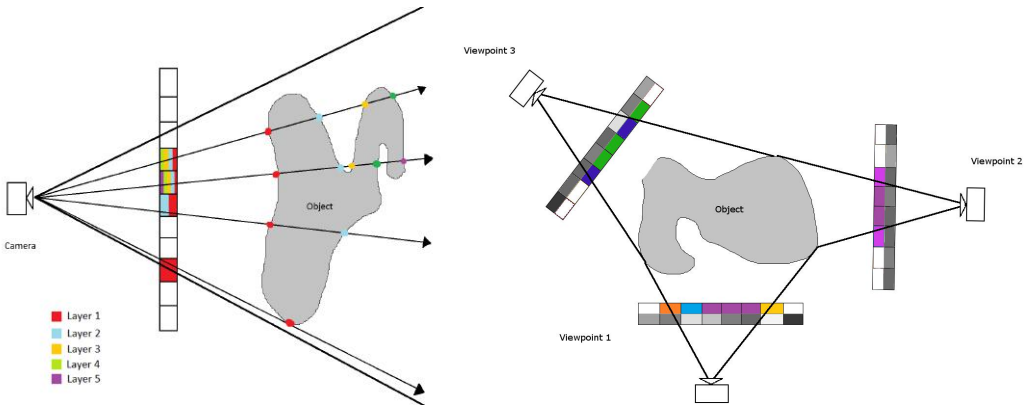
3.1. Early Results

For the early implementation we opted to use a simplistic representation of data to have a realistic baseline for comparison on further representation choices. The initial approach was to reconstruct the data into a separate point cloud for each captured frame. Each point cloud was saved to a file with (x, y, z) coordinates, and RGB color values for each reconstructed point. For the Kinect 1 device where the image resolution is 640×480, we have 307200 points, since each depth pixel represents a 3D point.

On Kinect 2, although having better depth precision, the depth image resolution is lower, standing at 512×424, at a total of 217088 points for each frame. Using the minimal data format possible, which is storing a long for the depth value (4 bytes in C++), and four bytes for the color (RGBA), we need 8 bytes per point, reaching a total of 1.74MB per frame, a total of 52.1MB per second at 30 frames per second, 3.13GB per minute of recording data. This is considering one single viewpoint. On our usual setup of 4+ cameras we have a minimum of 12.5GB of data per minute. With this large flow of data it is not only unfeasible to read every frame from disk at real time, but also goes over the amount we can store in RAM for instant reproduction.

Using the segmentation technique introduced in Section 4.2, we separated the representation of the static content from the one human actor in scene, keeping only one point cloud for the static content. With one person caught in video only, we used only 16.12% of the original space, with an average of 280KB for each point cloud, 8.40MB per second and 503.92MB a minute per each point of view.

Despite already being a significant improvement to the uncompressed scenario, they



(a) Layered depth image (b) Multiview plus Depth

Figure 3.2: Two different image-based representations based on depth.

still form a too large flow of data to be handled by personal computers. Assuming these values would scale linearly with the amount of dynamic elements on the scene, number of input devices, and also further improvements or processing of the data that might naturally increase the number of points, even on this best case scenario, a minute worth of data would use more space than four hours of high quality compressed video on a common codec. Some crucial points and bottlenecks on this simple representation were identified by these early attempts:

Point information redundancy: This was partially addressed by the simplification of the static elements to a single point cloud, but still persists on the dynamic elements that remain static for a good portion of frames meaning that from frame to frame there is a big amount of repeated information. This will be addressed through video compression techniques applied to the image-based streams 3.4.

Multi-stream integration: In order to achieve a representation that does not scale linearly with the number of sensors we use on the capturing process, we need to address the redundancy of data that is seen from more than one input stream. We want quality to improve, but not complexity. Our proposed data representation proposed in the following sections will address this problem.

3.2. Multiview Layered Depth Images

This work section is based on the work published in the Journal of WSCG Vol. 25(2), pp. 115, and presented at the WSCG 2017 conference. [1]

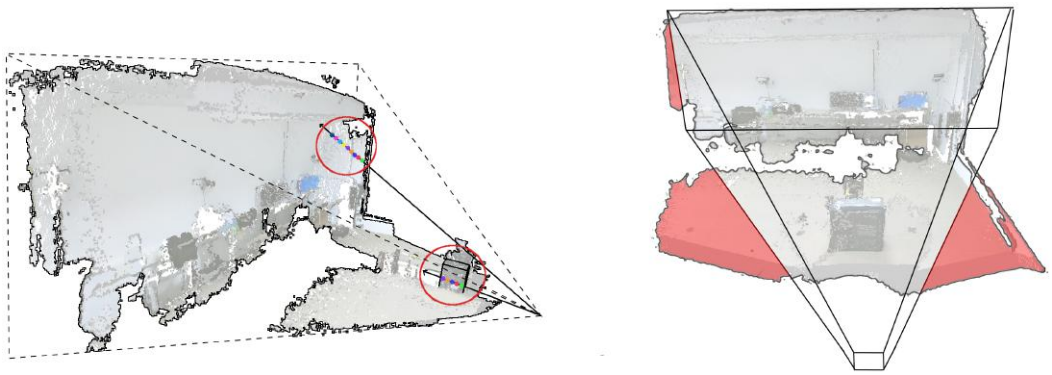
[1] Anjos et al. (2017)

Multiview+Depth (Figure 3.2b) encoding and Layered Depth Image Figure 3.2a) (Shade et al., 1998) have been the most popular approaches in the VBR scenario, with LDI being a reportedly less costly representation (Yoon et al., 2005). They are popular in this scenario due to the fact that they allow incorporating the continuous advances in video compression algorithms, being image-based.

However, both of the proposed representations are targeted for applications with a predefined user paradigm (3DTV, head-face parallax), and a preferably narrow-baseline capture setup. Advances in 3D capturing technology enabled developers to more accurately represent the real world, enabling less restrictive applications to emerge (Orts-Escolano et al., 2016) mainly in the fields of virtual and mixed reality. While recent work has been focused on coding tools for MVD, which can be applied to these new scenarios, we consider an alternative representation for a single frame, enabling higher compression from the start of the process.

The key advantage of the LDI representation over its alternatives is the fact that redundancy between views can be minimized during the encoding process (Kirshanthan et al., 2014). However, its classical representation of a central + residual viewpoints establishes a main viewing direction in which data can be optimally visualized, having disadvantages in a free camera navigation scenario. This has a direct effect in the redundancy computation, meaning that the sampling rate of the data is lower in the direction of the optical rays coming from the central viewpoint (Section 3.2.2.1). Moreover, on wide baseline scenarios where cameras might be in an opposite direction to the central viewpoint, it might not be possible for all data to be captured by one chosen central viewpoint (Figure 3.3b), requiring a more distant virtual viewpoint to be generated which, besides not being a trivial task, will decrease the sampling rate of the data. Lastly, surfaces that are quasi-parallel to the central viewpoint optical rays are intersected once in each layer, increasing the number of low-populated residual layers in order to encode the whole dataset (Figure 3.3).

We propose a novel image-based representation and encoding algorithm as a way to successfully answer the mentioned problems: the Multiview Layered Depth Image (MVLDI), where each layer is encoded according to a different viewpoint among the several capturing positions. We use a modified view-generation algorithm, which has better redundancy detection and a smaller number of generated layers, while being able to correctly encode data for a wide-baseline scenario, which was previously only possible using MVD and no redundancy estimation. We evaluate our technique with different datasets and camera setups and redundancy detection techniques.



(a) Surfaces parallel to an optical ray create low populated additional layers

(b) Choice of LDI central viewpoint might not encode part of the data.

Figure 3.3: Problems with the classical LDI representation

The next section reviews related work on MVD and LDI, followed by a description of the MVLDI generation algorithm and redundancy detection. Finally, results are presented with an in-depth comparison between our approach and the current image-based representations.

3.2.1. Related Work

Merkle et al. (2007) and Smolic et al. (2008) introduced Multiview+Depth coding, where depth data is associated with the video and encoded as a video stream. Although compression artifacts can be found in the rendered results, authors claim intermediate views are more easily generated just at the user side with full data. More recent work by Do et al. (2012) propose an inpainting algorithm which has a fast GPU implementation where holes are filled with the average on a 5x5 neighborhood.

Recent developments in this area have been related to depth coding. How to avoid the loss of precision due to compression, and how to use inter frame relations between depth values make better use of predictive frames. The two main strategies are independent depth coding (Merkle et al., 2009), and texture-assisted (motion and structure shared) (Lei et al., 2015), which tries to correct misalignments between depth and color. Kim et al. (2015) propose a method to evaluate how depth coding influences the quality of the results, which can be used by other authors.

Merkle et al. (2016) introduces a plane fitting approximation to simplify blocks of data (segmented by contour). This contour-based segmentation can be used to extract meshes (using Delaunay triangulation on the edges of the contour) and to

simplify data by simplifying geometric information and have a better encoding of the data.

Despite allowing effective 3D representations, MVD coding implies dense representations not considering the redundancy always present when multiple cameras image the same scenario. The LDI concept has been introduced to allow saving data by reducing redundancy.

Yoon et al. (2005) applies the LDI concept for VBR, proving to be a more compact than standard multiview coding. The data size of the multiview video linearly increases as the number of cameras. Authors propose using the LDI representation to compress and transmit this data, mainly due to the fact that redundant data that is seen by more than one point of view is not transmitted. All optimizations applied to MVD can be applied to LDIs, while not including repeated points, claim that is supported by Kirshanthan et al. (2014). In their following work (Yoon et al., 2007), improvements to the LDI representation are proposed (layer aggregation and layer filling) so temporal coding has a better performance.

Muller et al. (2008) discusses about image-based representations in the context of 3DV systems, and head motion parallax as an interaction paradigm. Authors claim that for stereo video, V+D is enough. For several views, MVD where only a subset of views with depth would be transmitted and intermediate views synthesized at the receiver side. They then introduce a different concept of LDI that is focused on 3DV systems. Instead of transmitting all the views, they transmit a central view close to the desired by the user, and residual information from side views to correct errors.

More recent work is focused on proposing better encoding for residual layers. Daribo and Saito (2011) proposed a different residual layer estimation algorithm which includes inpainting and hole-filling. Also, Kirshanthan et al. (2014) proposes efficient encoding for this residual information including pre-processing on the data for easier layer generation.

We propose a novel image-based representation and encoding algorithm, which varies the viewpoint among the several capturing positions, in order to allow larger acquisition and visualization baselines.

3.2.2. Description

The LDI is represented by a set of M layers $L_{ldi} = \{l_1, l_2, ..., l_M\}$ with each layer being an image-based representation of the world as seen from the same chosen "central viewpoint" v_c among the set of acquisition viewpoints of the data $V = \{v_1, v_2, ..., v_N\}$.

We propose a different representation, where one MVLDI will consist in a set of M layers $L_{mvldi} = \{l_1, l_2, ..., l_M\}$ where each layer l_i is represented according to one of the viewpoints v_i in $V = \{v_1, v_2, ..., v_N\}$. Each viewpoint v_i has its own intrinsic and extrinsic calibration parameters in order to generate the layer information.

The number of layers M has no direct relation to the N in the case of the LDI, while on our approach, M is typically equal to N, only being higher in the case of camera calibration misalignment.

The classical representation for LDI has two disadvantages in wider baseline capture scenarios, as exemplified in Figure 3.3. Encoding of parallel surfaces to optical rays, and excluding data from the process due to the central viewpoint choice. This does not happen in MVD encoding, due to the fact that encoding is performed according to different points of view. No data is discarded since it is seen at least once by its original recording viewpoint, and underpopulated layers are not created by parallel surfaces due to the fact that each subsequent layer will have optical rays in different directions. Our proposed representation MVLDI combines the advantages of both approaches, removing redundant data, and correctly encoding wide-baseline scenarios.

3.2.2.1. Encoding algorithm

Our layer generation process is similar to the traditional LDI algorithm described by Shade et al. (1998). Recently, different processes have been proposed for this step (Daribo and Saito, 2011; Kirshanthan et al., 2014), which targeted optimizations for specific use case scenarios. The proposed hole-filling and in-painting techniques were targeted to a head-face parallax user interaction paradigm, where corrections can be made at image space, and assuming all pixels must be filled at all times. If such assumptions were done about the scenario to be used with MVLDI, these could be easily incorporated. However, we established a more general scenario, only assuming depth values were available per pixel, which is common nowadays using

commodity depth cameras (e.g. MS-Kinect, Asus Xtion Pro, Intel RealSense).

Initially, each of the RGBD frames is transformed into a point cloud using the u, v image coordinates of each pixel, the depth value $d(u, v)$, and the intrinsic parameters of the capture device. All resulting clouds are then combined in a single volume P, using the transformations in the extrinsic matrix. Algorithm 1 describes the encoding process for a given volume P.

Algorithm 1 MVLDI encoding algorithm

```
Input: Point Cloud P, Acquisition viewpoints V
Output: MVLDI M
L ← set of layers for each viewpoint
R ← empty cloud
t ← threshold distance
for all point p_i ∈ P do
  for all layer l_j ∈ L do
    d_ij ← worldToImageSpace( p_i, l_j )
    e_j ← l_j[ d_ij.u, d_ij.v ]
    if isInside( d_ij, l_j ) then
      if isRedundant( d_ij, e_j, t ) then
        break
      else
        if e_j is empty then
          addToLayer( d_ij, l_j )
          break
        else
          if d_ij.d < e_ij.d then
            replaceInLayer( d_ij, e_j, l_j )
            retryPoint( e_j )
            break
          end if
        end if
      end if
    end if
  end for
  addPoint(R, e_j) //not encoded in any layer
end for
if R is not empty then
  re run with R
end if
discard empty Layers
```

We create N data structures, one for each layer l_j, where N is the number of cameras in the capture process. Each one is positioned according to the extrinsic calibration parameters for each one of the viewpoints, and is sized according to the recording

resolution of the input devices.

For each input point p_i, we try to encode it in a layer l_j by projecting it to that layer image space as a depth pixel d_{ij}. If $isInside(d_{ij}, l_j)$, meaning that (u, v) coordinates of d_{ij} are positive and smaller than the width and height of the layer, we check for redundancy $isRedundant(d_{ij}, e_j, t)$ (described in Section 3.2.2.2) where e_j is the depth pixel in layer l_j at the (u, v) coordinates of d_{ij}. In the case the data point is redundant, we start processing p_{i+1}, marking p_i accordingly, not including it in l_j. Otherwise, the point is encoded if the depth pixel e_{ij} is currently empty. If it is not, and d_{ij} has smaller depth than e_{ij}, we add d_{ij} to l_j, and e_{ij} is added for re-encoding. If e_{ij} is already filled with a point with smaller depth than d_{ij}, we go to the next layer.

Finally, in the case the point can not be placed in any of the layers, we add it to a collection R which will be processed with newly created layers. This is only the case when calibration errors exist. In all of the tested datasets these points were always encoded to a single layer, and represented less than 0.1% of the original point cloud.

3.2.2.2. Redundancy detection

In the classical LDI definition, a point P_{uvd} is considered to be redundant if $\|P_{uvd}.d - L_i[u, v].d\| < t$. Due to the fact that a central viewpoint is defined, and only depth comparisons are made in this particular image-space, sampling of data is not equal in all directions. When parallel to the viewing plane, pixel distance is evaluated, which in world coordinates is higher proportionally to the depth value and focal lenght f. When along the optical rays coming from the LDI viewpoint, a fixed threshold value t is used.

On a head-face parallax or 3DTV scenario, this was not seen as a problem, due to the fact that visualization is meant to be parallel to the central viewpoint. The sampling rate in the viewing direction is not perceived in these scenarios due to the visualization position restrictions. The lower sampling rate in the background pixels is also not noticed due to the fact that a parallel movement to the central viewpoint viewing plane does not change their image-space distance, not revealing possibly empty pixels.

Algorithm 2 shows our approach to the same problem, and Figure 3.4 illustrates the algorithm. The first key aspect of our method is using world coordinates to estimate the distance between the pretended point and the correspondent point in the layer,

Algorithm 2 Redundancy detection algorithm. n surrounding pixels are checked in order to properly evaluate all points that might be under the threshold distance t.

Input: P_{uvd}, P_{xyz} point to be encoded in image and world coordinates
Output: true or false
$L \leftarrow$ current Layer
$t \leftarrow$ threshold distance
$f \leftarrow$ focal lenght from intr. matrix
$s \leftarrow \frac{P_{uvd}.d}{f}$
$n \leftarrow \frac{t}{s}$
for $i = P_{uvd}.u - n$ to $i = P_{uvd}.u + n$ **do**
  for $i = P_{uvd}.v - n$ to $i = P_{uvd}.v + n$ **do**
    $Q_{uvd} \leftarrow L[i, j]$
    if $\|Q_{xyz} - P_{xyz}\| < t$ **then**
      return true
    end if
  end for
end for
return false

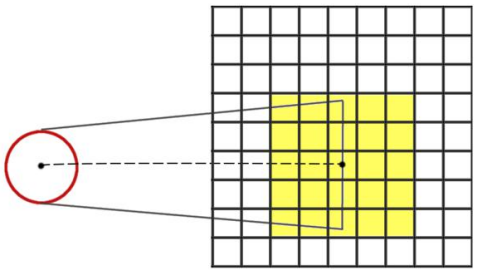
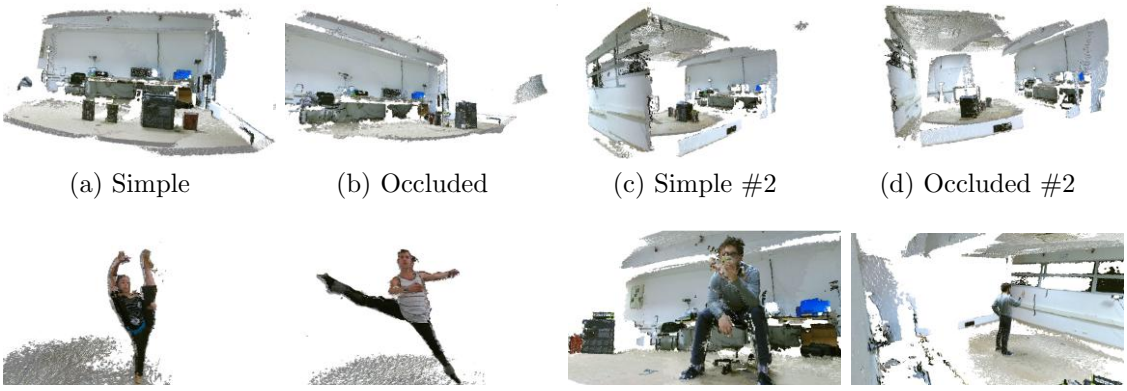


Figure 3.4: Global threshold illustrated

opposed to just the depth value. The further away from the central viewpoint of an LDI, the greater the discrepancy between a depth-based threshold, and one based in euclidean coordinates.

The second aspect is the fact that we also consider surrounding pixels to P_{uvd}. We first calculate the world distance s between pixels at that depth by calculating $\frac{P_{uvd}.d}{f}$ where $P_{uvd}.d$ is the depth value of the point being analysed, and f the focal distance from the intrinsics matrix. We then calculate the number of pixels to be checked around the encoded point by dividing the threshold t by s. By doing so, we very efficiently check every possible point that might be in a distance smaller than t from the considered point. Further layers do not need to be checked since the order of layer consideration is the same for each point, so in the case of a point not being



(a) Simple (b) Occluded (c) Simple #2 (d) Occluded #2

(e) Dancer F (f) Dancer M (g) Sitting (h) Selfie

Figure 3.5: Snapshot of our used datasets. Simple #3 is the same as Simple #2 but having 3 more cameras.

considered redundant in that layer, we will naturally move to the next one to perform the same test.

Name	# Cameras	Baseline	Pt. number
Dancer M	3	Wide	68.932
Dancer F	3	Wide	75.146
Simple	4	Narrow	563.315
Simple #2	4	Wide	491.707
Simple #3	7	Wide	828.822
Occluded	4	Narrow	511.161
Occluded #2	4	Wide	505.232
Sitting	4	Narrow	580.736
Selfie	4	Wide	511.161

Table 3.1: Description of the tested datasets.

3.2.3. Results

We tested our approach with varied datasets captured with the Microsoft Kinect 2 sensor, each one holding different properties. Table 3.1 gives a brief description of each dataset, and Figure 3.5 shows a snapshot of each. These datasets were captured either in a controlled laboratory scenario, or are single frames of the capturing sessions with Rui Lopes Graça with Killian Souc and Miyu Matsui, described in Section 1.1. We compared MVLDI with LDI using both the proposed global thresholding algorithm, and the traditional image-based technique in order to individually evaluate the effect of each contribution. MVLDI with global thresholding represents our

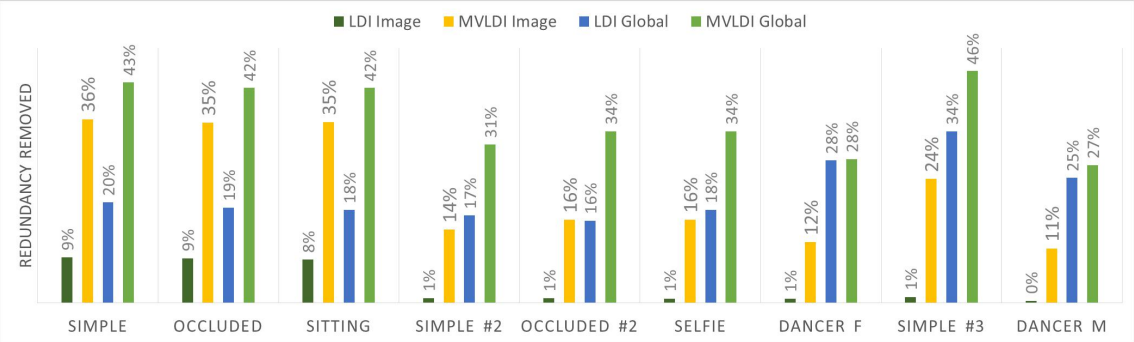


Figure 3.6: Percentage of data detected as redundant from the original cloud.

proposed technique, and LDI with image-based thresholding, the traditional LDI implementation. Our goal was to minimize the number of generated layers, while more effectively detecting redundant data.

Both narrow and wide baseline were contemplated in order to validate our claim of MVLDI having a superior performance than LDI on a wide scenario. Different numbers of acquisition devices were also tested in order to validate the scalability of the process, and also validate the claim that redundancy is proportional to the number of capture devices. Different datasets contained different distribution of points according to their content. With the used devices (Kinect one), each stream can contain up to 217088 points. Empty areas in the datasets are typically due to limited distance where the sensor works (e.g. Simple with 563.315 points, 64% of the possible total for 4 sensors).

Finally, while most of our datasets were controlled lab scenarios in order to control the disposition of the objects related to the cameras, we also included data captured from a realistic dance scenario (Figures 3.5e 3.5f). "Simple" datasets contained lines of boxes visible by most of the cameras (Figures 3.5a 3.5c) , "Occluded" datasets had a line of boxes occluded in one of the points of view (Figures 3.5b 3.5d). Sitting and Selfie (Figures 3.5g 3.5h) included a person in a controlled scenario in order to better evaluate if redundancy detection had a negative effect in the perception of more delicate shapes.

Dancer M 0%	Dancer F 0%	Simple 19%	Simple #2 48 %	Simple #3 39%
Occluded 19%	Occluded #2 46%	Sitting 20 %	Selfie 45%	

Table 3.2: Percentage of points discarded by the LDI approaches due to being outside of the frustum of the central viewpoint.

3.2.3.1. Quantitative Analysis

Figure 3.6 shows the achieved results regarding redundancy detection. The values in the table are calculated regarding the total number of points considered for encoding. Table 3.2 shows the percentage of the total dataset that was discarded in the case of LDI due to being outside the viewing volume of the central viewpoint.

Our approach had a superior performance over LDI in all tested datasets. Notably, the LDI approach had a high number of low populated layers (over 100 in Simple, Occluded #2 and Selfie, as seen in table 3.3) This experimentally confirms the problems exemplified in Figure 3.3a. With a wide-baseline capture setup, surfaces perpendicular to the central viewpoint viewing plane will create an elevated amount of created layers. Also, redundancy detection on the wide-baseline scenarios was lower with the classical approach. This is related to the amount of non-encoded points (over 40% of in some scenarios (Table 3.2, example in Figure3.3b) which would include walls and floor sections where typically a lot of redundancy was found, but also due to the redundancy detection algorithm.

The proposed global thresholding technique performed better in all scenarios. The difference was smaller in narrow scenarios, since the data was captured and sampled from the same perspective, so a pixel-based comparison still had an impact, albeit smaller. The biggest difference was found in wide baseline scenarios, where several points under the used threshold were located in neighboring pixels but not considered redundant only considering the depth value.

When comparing solely the difference in the number of points of view, the multiview approach had a smaller number of layers in all scenarios, being close or equal to the baseline for comparison (MVD, where the number of layers is equal to the number of input devices). The multiview approach did not suffer from the problem presented in Table 3.2, where big segments of the point cloud were discarded due to not being visualized by the central viewpoint. This is essential in a free visualization application through a virtual camera, specially in wide-baseline scenarios. Redundancy detection in MVLDI was higher in all scenarios except for the Dancer scenarios where only three cameras were used, and no data was left out of the encoding process.

An argument could be made about the choice of the central viewpoint for the LDI, and considering a virtually generated point of view that would include all of the data. The main problem with this proposition is the fact that a single viewpoint that includes all the data would necessarily be placed further away than the existing

acquisition viewpoints, which certainly increases the problem in Figure 3.3a. The further the viewpoint is from the data, the more likely the planar surfaces in the cloud are aligned with the optical rays, increasing the number of low-populated layers.

Several previous works on LDI have reported higher rates of redundancy detection than the ones presented in this section (Yoon et al., 2005). The datasets typically used for benchmark are the breakdancers and ballet sequences from Microsoft Research, which are aimed to a head-face parallax interaction, or 3DTV. Cameras are separated by approximately 20cm from each other, which is a very narrow baseline. Also, depth precision is considerably lower, with only 256 values to represent the whole range of the scene. The depth cameras used in this work have a precision in the order of milimiters, which is why less values with the same depth are encountered using the image-based algorithm.

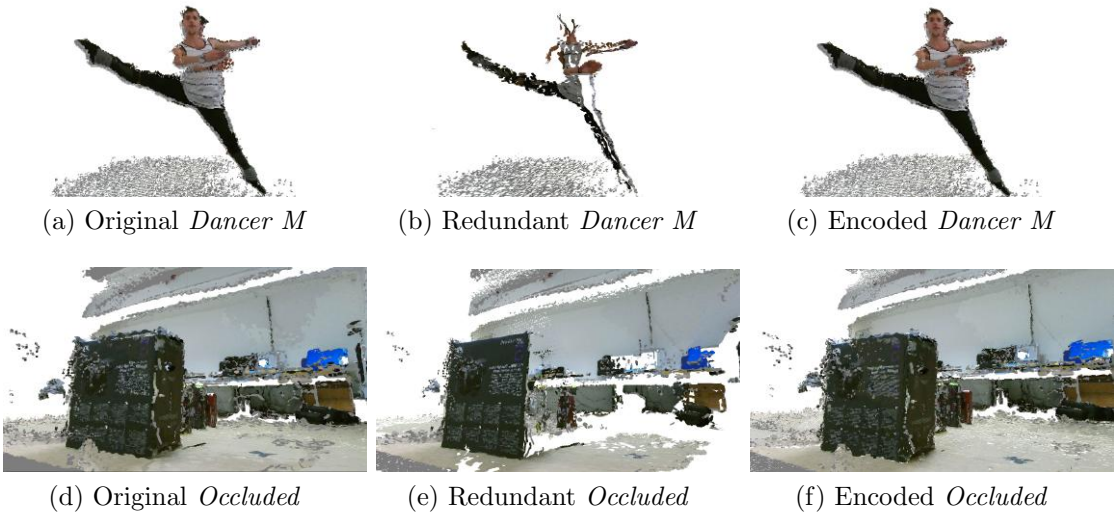
3.2.3.2. Qualitative Analysis

Although we report high redundancy removal in all of the presented scenarios, the quality of the visualization was not compromised. Figure 3.7 shows a side by side comparison of the input cloud, the redundant data, and the encoded result. On the dancer example where 27% redundancy was reported, data from the dancers back, silhouette, and floor were correctly reported as redundant (Figure 3.7b), not compromising the final visualization, as seen in Figure 3.7c.

On "Occluded" where a narrow baseline setup was used, we can clearly see a large amount of data (42%) consisted of walls, floor, and only the front part of the non-occluded box as being reported as redundant (Figure 3.7e), with the remaining of the data, included the boxes behind the front one, being perfectly visible in the encoded version (Figure 3.7f).

Regarding precision loss during the LDI encoding algorithm, we kept it to a minimum by limiting the depth where the point is considered to be inside the current layer to the maximum depth where the used cameras give reliable values. This can be seen in the similarity between the input and output examples seen in Figure 3.7

Also, considerations about the viability of MVLDI on a video scenario can be made. Our generated layers are less sparse as seen on figure 3.8, and in a smaller number. We can more easily guarantee coherent matches between frames using block matching algorithms due to having more densely populated regions on the image. Also,



(a) Original *Dancer M* (b) Redundant *Dancer M* (c) Encoded *Dancer M*

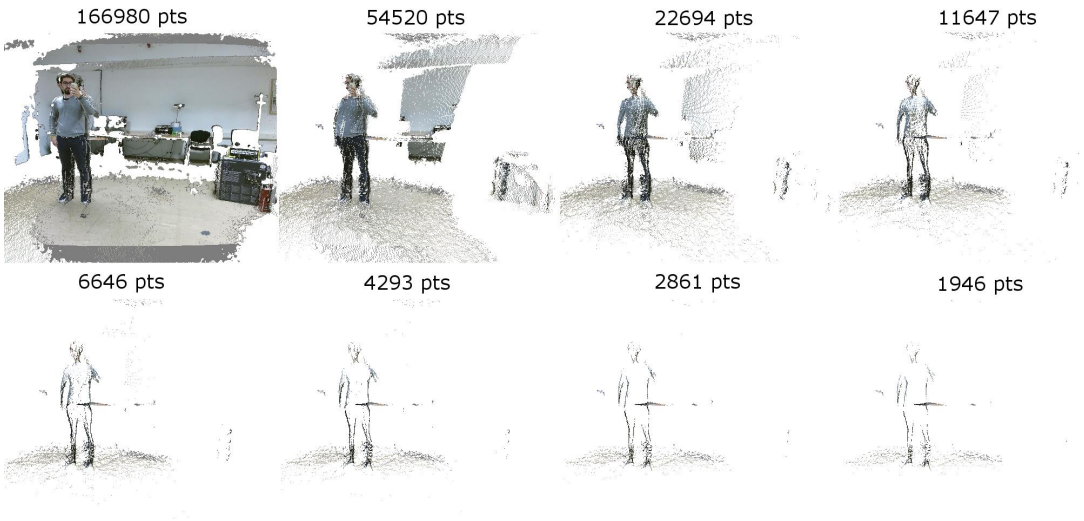
(d) Original *Occluded* (e) Redundant *Occluded* (f) Encoded *Occluded*

Figure 3.7: Result of the encoding process. Original (a, d), redundancy removed (b, e), and encoded data (c, f) for a wide and a narrow baseline scenario (top vs bottom row).

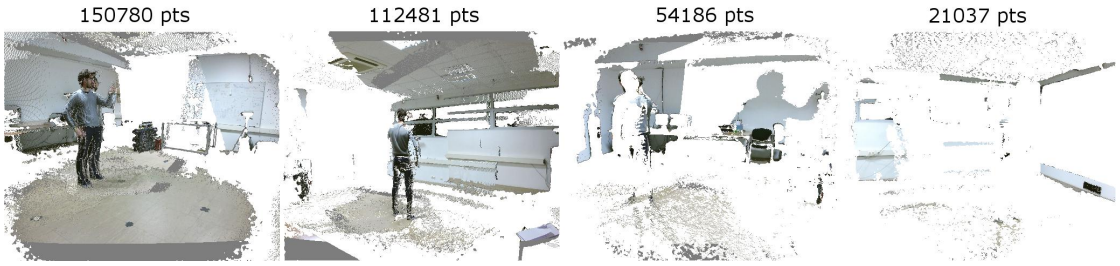
our final number of layers is typically equal to the number of acquisition viewpoints, unlike LDI's which are inherently dependent on the content of the scene, and might create uneven distribution of layers per frame, which are less reliable for temporal matching. When compared to MVD's, our representation discarded redundant data, creating more compact frames.

Name	LDI Image	MVLDI Image	LDI Global	MVLDI Global
Dancer M	23	4	7	3
Dancer F	20	4	6	3
Simple	10	5	8	4
Simple #2	115	5	11	4
Simple #3	88	11	16	8
Occluded	13	5	12	5
Occluded #2	119	5	10	4
Sitting	17	5	12	5
Selfie	125	5	9	4

Table 3.3: Number of layers generated by each approach. MVLDI with global thresholding has the overall lower number of layers.



(a) LDI



(b) MVLDI

Figure 3.8: Comparison between the first 8 LDI layers, and the full 4 MVLDI layers of the "Selfie" dataset, with number of points per layer. The silhouette of the subject, floor and a table in the background generate low populated layers being parallel to the optical rays.

3.3. MVLDI view generation

Although successful, the proposed MVLDI generation algorithm does not consider the impact of the evaluation order of the viewpoints. Given that each generated layer will be compressed into a separate video stream, the ideal output has as few layers as possible (Figure 3.9), so the decompressing and rendering process is simplified. Also, more densely populated layers are more easily matched between frames by block matching algorithms used in video compressors (Zhu and Ma, 2000; Yoon et al., 2007). Both of these aspects can be negatively impacted by a poor choice in the viewpoints evaluation order. Figure 3.10 describes these problems in detail.

In Figure 3.10a, if V_{p1} is chosen for evaluation before V_{p2}, the red portion of the point cloud would not be included in the first layer, forcing a second layer to be created. This would have been avoided by evaluating V_{p2} first. A similar scenario

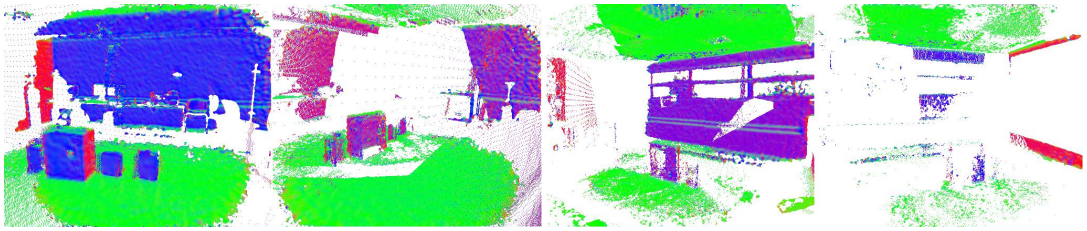
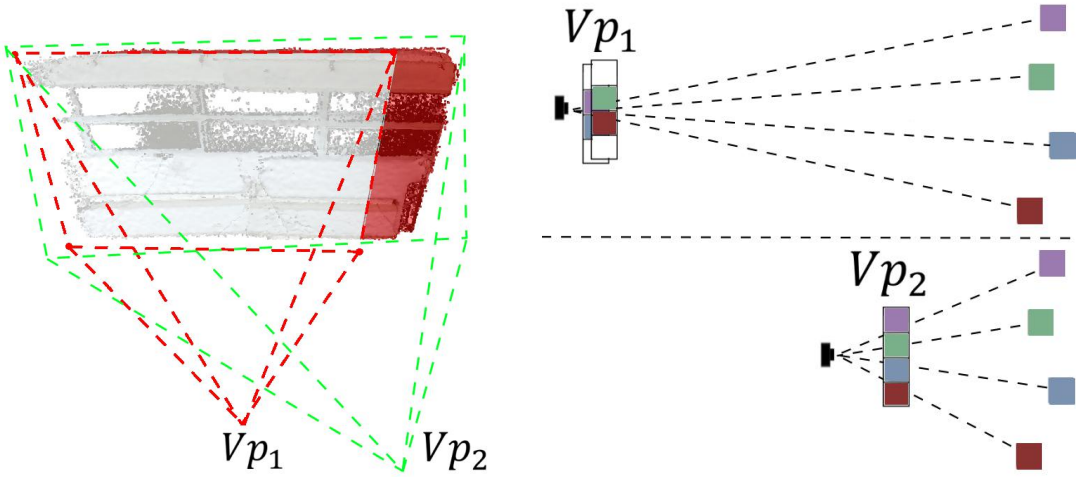


Figure 3.9: Sequence of optimal MVLDI layers, colored with the estimated normals



(a) First problem (b) Second problem

Figure 3.10: Problems with poor MVLDI viewpoint selection

can be seen in Figure 3.10b where four non redundant points are being encoded. While V_{p1} can only include two of those points at each time due to its distance, V_{p2} can create a single layer with all points.

This section describes and evaluates five different techniques for viewpoint selection on MVLDI. We implemented and tested five (plus one baseline) different approaches comparing to two different baselines: a randomized approach and a full complete search by choosing the best possible order of evaluation according to our set criteria. Our results show that the greedy search-based approach has the overall edge over the cluster-based ones, due to the sparsity of the residual data in the last layers which creates problems for clustering algorithms. However, our results also showed that normal-based viewpoint selection and generation can be very successful in more complex wide-baseline scenarios.

A synthetic viewpoint generation approach is presented which can be used not only when viewpoint information is not available but also showed better results when increasing the number of input cameras, being applicable in different scenarios than VBR, where viewpoint information is not available.

A short review of related work will be presented, followed by the description of the implemented algorithm, evaluation and results and concluding with guidelines and future work.

3.3.1. Related Work

Although our work is focused on a specific part of the MVLDI algorithm, there are several different aspects worthy of attention regarding image-based representations for VBR. This section will review Point Cloud analysis techniques. For related work on image-based representations for VBR, please refer to Sections 2.5.2.2 and 3.2.1.

One possibility when considering an informed approach on choosing encoding viewpoints for a point cloud is to look for information on the data being encoded. Given that the point cloud is an unstructured data type, different techniques need to be applied to extract information from them. Pauly et al. (2003) propose different features for point clouds in the shape of feature curves, which are useful for detecting contours and recognizing repeated objects in a point cloud. Segmentation has been used as a way to detect surfaces, objects, and regions that share properties.

The survey by Nguyen and Le (2013) describes the different approaches to point cloud segmentation, discussing their advantages. Notably, Rusu and Cousins (2011b) have implemented different attribute based methods into the open source project "Point Cloud Library". Normal vectors, position, color, and curvature are examples of attributes used in the segmentation process More complex features as the ones mentioned above (Pauly et al., 2003) can also be used for segmentation in some scenarios.

3.3.2. Algorithms

We implemented six different approaches to the viewpoint selection problem, with one (optimal search) being used as a baseline (together with arbitrary attribution, as in the original paper). This section will describe each new approach in detail.

3.3.2.1. Search-based Selection of Viewpoints

Our goal is to create as few layers as possible, while being densely populated. These approaches take into consideration the layer depth and number of points in order to find the best order of evaluation.

Optimal

Algorithm 3 describes the process of creating the encoding tree with all possible combinations. For each viewpoint V, we try to encode all points $p_i \in P$, being P the complete point cloud. A tree node is created with the the number of encoded points n_p. If this value is positive, we go deeper in the tree by running the process again with the non-encoded points R. If this value is zero, we create a "dead end" node, representing that the current viewpoint choice did not visualize a single point. If R is empty, it means a solution was found.

Algorithm 3 Encoding tree creation algorithm for optimal search

```
Encode Optimal
Input: Point Cloud P, Acquisition viewpoints V
Output: Encoding Tree T
R ← empty cloud
T ← empty tree
for all viewpoint v_j ∈ V do
  for all point p_i ∈ P do
    Layer l ←Create Layer(v_j)
    if encodePointLayer(p_i, l) == false then
      addPoint(R, p_i)
    end if
  end for
  N ← createTreeNode(n_p)
  if n_p > 0 then
    if R not empty then
      N.childNodes ← Encode Optimal(R, V)
    end if
  else
    N ←"dead end"
  end if
end for
```

After the encoding tree T is created, we find the minimum depth where a solution was found, pruning the rest of the tree. Then, we traverse the tree bottom-up comparing at each level the number of total points (current node + child nodes), choosing the maximum. After we have the best solution, Algorithm 1 is run with V populated with the correct order of viewpoints.

This algorithm is one approach to maximize the set objectives. However, its complexity can be estimated to be $O(v^d)$, where v is the number of viewpoints and d the maximum depth set for the tree creation. This is very high considering that the number of points to be considered in each encoding step also scales linearly with v and that this process might be executed several times in a video sequence. On a scenario with 7 cameras we can go easily beyond a billion nodes in the encoding tree, with a high cost on creating each one of them.

Greedy

Due to the complexity of the previous approach, we implemented a greedy approach where instead of creating the full tree at the beginning, we simply choose the best option at the moment. At each step, the point cloud is encoded to different layers created according to the viewpoints v_j in V. The one that contains the highest number of points n_p is added the the result MVLDI. This process is repeated until no points are left. If at any point a certain viewpoint creates an empty layer, it is removed from V in order to simplify the following steps. Algorithm 4 shows this approach in more detail.

Algorithm 4 Greedy encoding approach

```
Encode Greedy
Input: Point Cloud P, Acquisition viewpoints V
Output: MVLDI M
R ← empty cloud
L ← list of layers
for all viewpoint v_j ∈ V do
  l_j ← Create Layer(v_j)
  encodePoints(P, l_j)
end for
M ← l_j ∈ L with highest n_p
Remove v_j that created empty layers from V
if R is not empty then
  Encode Greedy(R, V)
end if
```

3.3.2.2. Iterative Clustering to Select or Generate viewpoints

Alternatively to selecting viewpoints, as suggested in Section 3.3.2.1, one may consider an informed approach that chooses the next encoding viewpoint according to the data at hands. In this scenario, we needed to define what attributes would lead us to a positive result. We chose to base our approach on normal-based clustering

and segmenting surfaces from the point cloud, and using this average normal value and center of mass to choose an encoding viewpoint. Looking into the positive results from the previous approaches, large surfaces with the normal vector pointing at the camera are typically not occluded, creating dense layers. Also, this will lead us into a higher probability of including pieces of the same element in a single layer. This would be advantageous in a temporal compression scenario, i.e. the object will move as a whole in that layer.

Algorithm 5 Cloud segmentation and parameter estimation

```
Input: Point Cloud P, Min. cluster size m, Incremental angle α,
Output: Biggest cluster avg. normal vector n and center of mass c_m
C_max ← empty cluster
θ ← 2
while θ < 360 do
  Clusters C = SegmentCloudByNormals(P,θ)
  C_max ← FindBiggestCluster(C)
  if C_max.size > m then
    break
  else
    C_max ← empty cluster
    θ ← θ + α
  end if
end while
if C_max is empty then
  C_max ← P
end if
c_m ← estimateCentroid(C_max)
n ← estimateAverageNormal(C_max)
```

For all three implemented techniques, we used the same approach to estimate clusters and average normals, which can be seen in Algorithm 5. We define an initial threshold angle θ on the difference of normals accepted in the cluster (experimental results started at 2 degrees), and repeatedly apply the segmentation process using the conditional region growing method in the work of Rusu and Cousins (2011b). We define the minimum size for a cluster m, based on the type of data and depth cameras used (for our setup 3% of the size of the total point cloud). If no clusters biggest than m are found, θ is incremented by a fixed facto α so the clustering condition is incrementally relaxed, until clusters are made only based on their distance. If no clusters are found, or the size of the total point cloud is smaller than m, we define the whole point cloud P as being the biggest cluster. Then, we estimate the center of mass c_m for the point cloud, and the average normal vector n to be used in the following algorithms Finally, Algorithm 6 describes how all of the

following approaches create an MVLDI, differing only on the *chooseViewpoint(P)* method.

Algorithm 6 Encoding algorithm for iterative clustering approaches

```
Encode Optimal
Input: Point Cloud P
Output: MVLDI M
R ← empty cloud
T ← empty tree
while P not empty do
  V ← chooseViewpoint(P)
  Layer L ← encodeToViewpoint(P)
  M.add(L)
end while
```

Best Direction The first algorithm using this information is focused only on the resulting n vector of Algorithm 5. Our objective is to find the surface that most directly face a certain $v_i \in V$. At each step of the encoding process (Algorithm 6), we calculate $n.f_{v_i}$ being f_{v_i} the front vector of the viewpoint $v_i \in V$. The dot product represents $cos\phi$ being ϕ the angle between the two vectors, with 0 meaning the vectors are perpendicular, 1 parallel, and -1 exact opposites. In our system, we estimated the normals to the point cloud pointing away from its center of mass, consequentially facing away from the camera that captured that portion of the data. In that case, we select the viewpoint $v_i \in V$ which $n.f_{v_i}$ produces the higher value. We also considered a variant of this approach where we use the absolute value $|n.f_{v_i}|$ instead, so on wide-baseline scenario two opposed viewpoint are considered as equally favorable. These approaches are identified in Section 3.3.3 as *best_dir* for the first one, and *best_dir_2way* for the second.

Viewpoint Selection and Generation for the Multiview Layered Depth Image

Using the output information of Algorithm 5, we developed an approach that create synthetic viewpoints that will try to optimize the concept introduced in the previous section (viewpoints directly facing a surface). We define a viewpoint v_s along the ray starting in c_m following the direction defined by n. For this approach, we wanted the whole cluster that generated this viewpoint to be seen, thus the majority of its data would be encoded.

The distance d where the viewpoint v_{opt} is placed can be calculated by $\frac{size}{2\tan fov/2}$, as seen in figure Figure 3.11. By calculating the bounds $u_{max}, u_{min}, v_{max}, v_{min}$ of the cluster on image-space, we fit the camera to the biggest dimension, by calculating

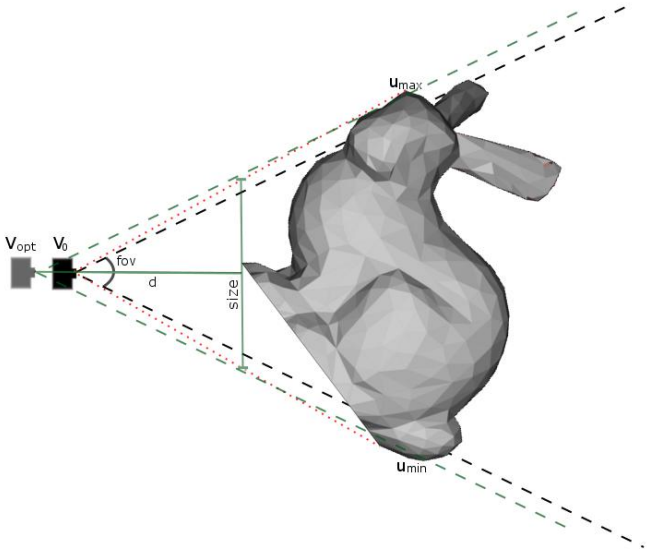


Figure 3.11: Distance where the synthetic viewpoint is placed on the viewpoint generation approach.

its respective *size* in world-coordinates, which will allow us to estimate the distance d where we want to place the camera in order to see the whole cluster. If no cluster bigger than the minimum size m is found, the whole cloud is used for this purpose.

Best Position

This approach follows the same process as the previous (Viewpoint Generation). However, instead of using the generated viewpoint, its position is used to choose the closest viewpoints among the original capture viewpoints (using the euclidean distance).

3.3.3. Evaluation and Results

We evaluated the implemented approaches using the nine different datasets used in the MVLDI paper (Anjos et al., 2017), which were captured with the Microsoft Kinect. The datasets were chosen due to the fact that they cover different type of content (different arrangement of simple objets, and also more complex objects), baselines and number of input cameras. The details of each point cloud can be seen in Table 3.1.

We selected the approaches previously described (*optimal_greedy*, *best_dir*, *best_dir_2way*, *generate* and *best_pos*) for comparison, setting the *optimal* approach as one baseline due to its elevated complexity, and also an arbitrary choice of non repeated view-

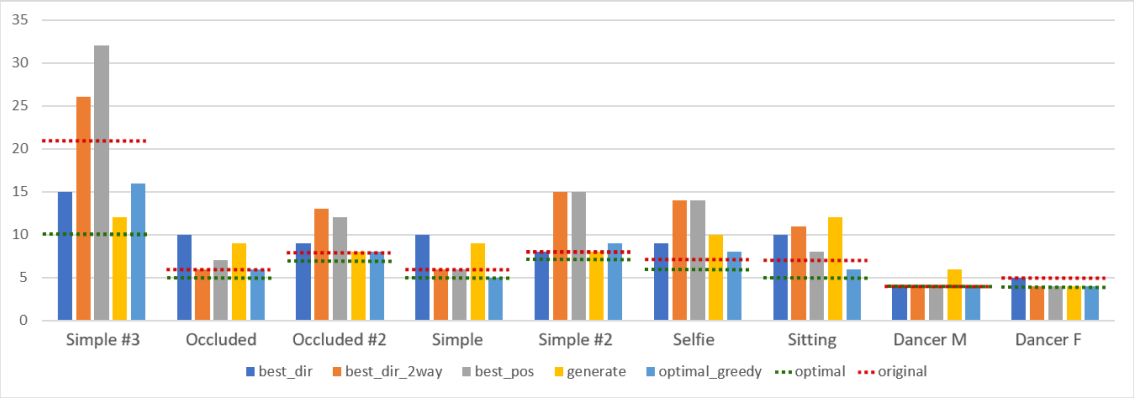


Figure 3.12: Layers generated for each approach/scenario

points as in the original MVLDI algorithm (Anjos et al., 2017) named *original* in the figures.

The goal of the experiment was to evaluate each approach according to the number of generated layers, the distribution of points per layer, and how different baselines and number of input cameras would impact the performance of each technique.

3.3.3.1. Number of Layers

Figure 3.12 shows the number of created layers for each approach/scenario combination. The optimal result is highlighted with a green dotted line, while the original approach with a red dotted line. Ideally, results would be as close as optimal, while being an improvement from an arbitrary assignment of viewpoints.

When we have a low number of cameras (3), viewpoint selection has a lesser impact on the number of generated layers, given that there are not a lot of possible combinations. The optimal approach can be considered viable in this scenario, since it is not as costly to verify all possible orderings. Notably, *generate* and *best_dir* had an inferior performance in this scenario due to the fact that less reliable surfaces could be found.

Overall, *optimal_greedy* was found to be the most consistent approach. However, its performance was slightly worse in wide-baseline scenarios (such as "Selfie", "Simple #2" and "Simple #3). These scenarios are more prone to the existence of residual data due to occlusions, and this approach is focused in populating highly the first layers without considering the impact in the next layers.

Cluster-based approaches were all affected by the same problem: when clustering residual data, the detected surfaces and estimated average normals are not indicative

of coherent parts of objects and surfaces as in the original data. This forces the algorithm into making poor decisions.

The approaches *best_dir2way* and *best_pos* have very similar results, with the approach *best_dir_2way* having the edge on most situations. Both have poor performance in wide-baselines, choosing viewpoints targeted at low populated areas due to normals indicating an opposing direction than expected. *Best_dir* suffered from the clustering problem, but still presented positive results in some cases, outperforming greedy in "Simple #2" and "Simple #3". This confirms the assumption introduced in Section 5, of choosing viewpoints pointed at big surfaces to create dense layers.

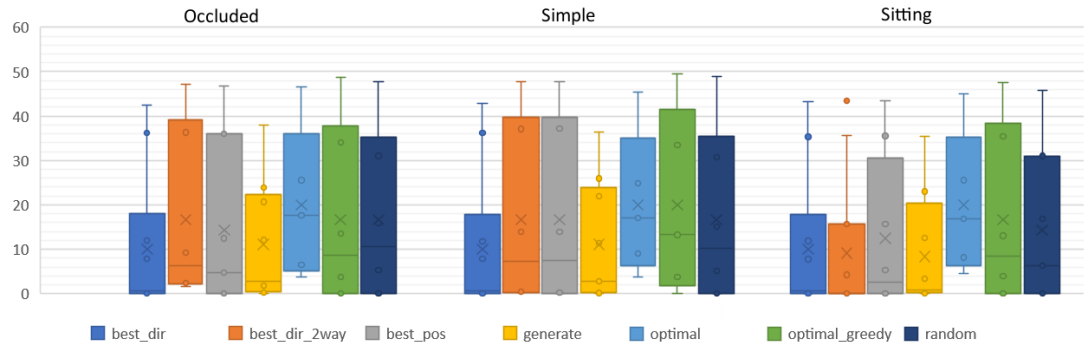
Generating virtual viewpoints was seen to be more advantageous in wide-baseline scenarios, but was only seen as the best approach in the "7-Wide" scenario. It is important to notice that this approach would typically discard more points due to redundancy than the others. This is due to the synthetic viewpoint being created further away than the original viewpoints, causing more precision loss to happen.

3.3.3.2. Points per layer

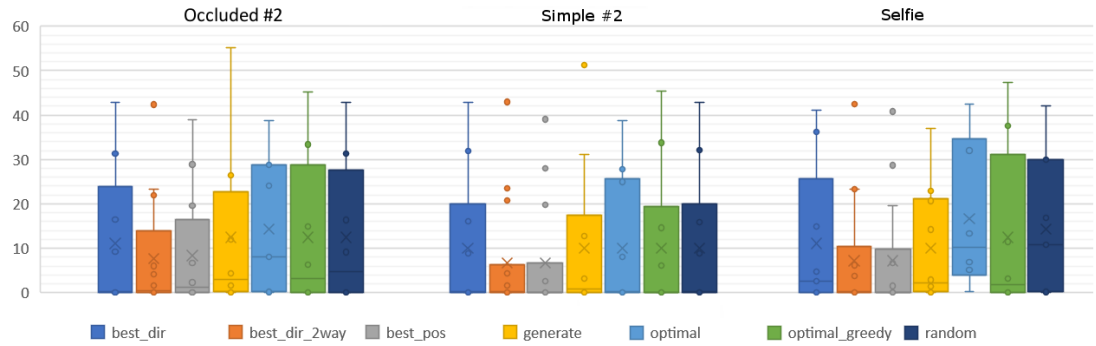
Figure 3.13 shows the percentage number of points per layer in a box plot. Whiskers and in some cases outlier points represent the maximum and minimum percentages, boxes represent the first and third quartiles, "X" marks the average, and the interior line the median value.

In the 3 cameras scenarios 3.13c, "Dancer F" had similar results for all approaches, with all of the proposed techniques outperforming the *original* approach. Notably, *generate* had the highest maxima due to sucessful detection of a cluster that was part of the dancer (Figure 3.14c). On "Dancer M", the clustering algorithm was not as sucessful, making all approaches fall slightly short of *optimal*, with *greedy*, *best_dir2way* and *best_pos* being the best alternatives.

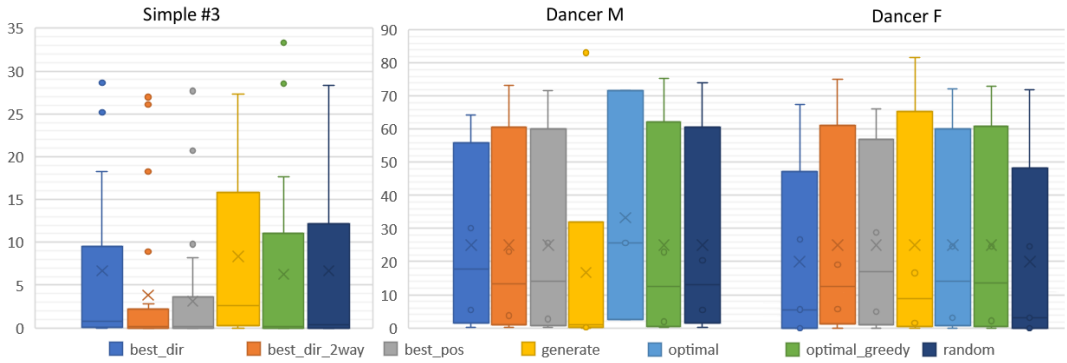
In the narrow baseline scenarios (Figure 3.13a), we can notice that *optimal* has higher minimum values than the other approaches. Once more, the *optimal_greedy* approach has shown positive results, with averages close to the optimal approach (Figure 3.14a), but also high third quartiles, which show that the second and third layers are also densely populated. Regarding the cluster-based approaches, minimum values and first quartiles were below 2% in all techniques, showing that a considerable number of low populated residual layers were created, given that they are not classified as outliers. Notably, *best_dir* had a poor distribution of layers as



(a) 4 Cameras Narrow baseline



(b) 4 Cameras Wide baseline



(c) 7 and 3 cameras wide baseline.

Figure 3.13: Percentage of points per layer

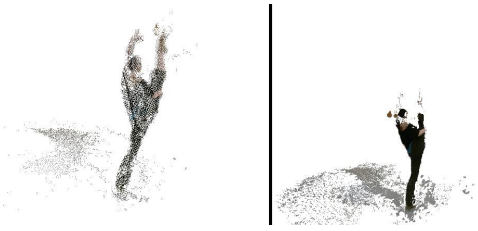
seen in the also low third quartiles seen in all three scenarios. This is coherent with the higher number of layers seen in Figure 3.12.

In the Four cameras wide-baseline datasets (Figure 3.13b, the poor performances of *best_dir2way* and *best_pos* are once again shown by the high number of outliers above the third quartile, specially in the "Simple #2" scenario. A high number of low populated layers were created in this approach (Figure 3.14b). *Best_dir* and *generate* had the best results form the cluster-based techniques, with similar averages and maximum values than the optimal approach, albeit lower third quartile values, revealing a less optimal distribution. *Optimal_greedy* had comparable results and average values to *optimal* and *original*, while typically having a higher maximum, and lower median.

Finally, "Simple #3" where 7 cameras were used, the *optimal* approach was not able to output the best possible distribution due to the high number of operations required, but only the depth where this solution would be created, so it was excluded from this comparison. *Generate* has showed very positive results in comparison to its alternatives, with no outliers, and higher median, average, first and third quartile values. *Greedy* and *best_dir* with similar values overall, with *greedy* having higher highs, at the cost of having one more layer Figure 3.12

(a) First two layers of greedy and optimal approach on "Sitting". Highly populated layers, with complete objects

(b) First two layers of three direction based approaches on "Sitting". Creates highly populated layers, but fragments the main object leaving several outliers.



(c) Generated viewpoint x Optimal original viewpoint. Synthetic viewpoint is closer to the object, having higher resolution.

Figure 3.14: Generated layers examples

3.3.4. Discussion

Combining both of the analyzed results, no technique was able to achieve the results by the *optimal* approach in all scenarios. However, *optimal_greedy* showed an overall more consistent behavior, having an advantage over an arbitrary attribution in all scenarios, and being near the optimal result in most scenarios. However as mentioned before, it showed worst results in wider baselines in both aspects, being specially bad with 7 cameras. On one side, this tells us that a search-based approach is viable. While creating the full tree is not possible in useful time (such as in the *optimal* approach), a greedy search does not consider any alternatives. Either a more informed heuristic, or allowing the algorithm to explore more than one node at each level, would possibly avoid overpopulating the first layers, which creates sparse residual layers.

As mentioned before, the main problem with the cluster-based approaches was the fact that coherent surfaces cannot be found in the residual layers, pointing the algorithm in a less optimal direction. An alternative viewpoint estimation method needs to be developed for residual layers which does not rely on surface detection, or different clustering conditions need to be used in the last portions of the data.

Even though these problems exist, *Best_dir* and *generate* outperform both the *original* and *optimal_greedy* approach in the wide-baseline scenario with 7 cameras, making an argument for their scalability in more complex scenarios. As seen in Figure 3.12, the importance of viewpoint selection increases with the number of used cameras. With more redundant data from different cameras (as seen in Section 3.2), higher the possibility that a portion of the data might be better encoded in a different viewpoint than it was originally seen.

When comparing *best_dir* and *best_dir_2way*, the first had the overall better results, confirming our approach of using the direction of the normal vector on the biggest clusters to select an encoding viewpoint. An argument for *Best_pos* to be used can be made on scenarios where the setup has cameras with the same orientation but different positions. In typical wide or narrow baseline setups, its performance was sub-par.

Finally, although the view generation approach (*generate*) was not found to be the best in all scenarios due to the clustering problems previously mentioned, it allows one to encode point-clouds into a layered representation without previous information of the capture viewpoints. When comparing the number of created layers with the tests performed with the LDI in the MVLDI article (Anjos et al., 2017),

where a single central viewpoint was used, the *generate* approach has a considerably lower number of layers. This approach can be used with complex static point cloud scans scenarios in the context of Image-based rendering, specially datasets created by sweeping scanners, where the other approaches could not be used.

3.4. Video compression

In order to further evaluate our approach and its applicability to the VBR scenario, we implemented and tested video compression applied to our single frame representation. This is typically done in VBR works that use an image-based representation, since as ordinary video, an uncompressed lengthy recording is typically not usable in video players. Known algorithms with open source implementations were used so further developments in single-frame representations can easily be compared to our approach.

Color information was compressed using the h.264 algorithm (Richardson, 2004), which is implemented in the NVIDIA video codec SDK (NVEnc) [2]. This implementation uses CUDA to speed up the compression process, which compensates the extra steps taken to generate each frame. More importantly, the faster decompression speed allows us to achieve interactive frame-rates during the rendering process.

In order to more easily apply the rendering process described in Section 4.3.2, we also compressed the estimated normal vectors from the point cloud into a separate color stream. We directly map the xyz coordinates in the rgb color space, and use the a value to keep a bit mask of the signal of each coordinate:

$$a \boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{sig_x}\boxed{sig_y}\boxed{sig_z}$$

Finally, the depth stream was encoded using the RVL algorithm proposed by Wilson (2017).

Wilson's compression scheme is a combination of run length encoding and variable length encoding compressing each frame independently using the process described in the paper. As mentioned by the author, lossy compression techniques are optimized for color images and do not support 13bpp or 16bpp formats. Encoding part of the depth value in each one of the color channels using a lossy scheme, creates errors in the channel holding the most significant bits, creating large artifacts in

[2] https://developer.nvidia.com/nvidia-video-codec-sdk, accessed october 2017

the reconstructed depth image. Although this approach does not use temporal coherence, it was reported by the author that its performance is superior to a similar approach encoding temporal differences. Another positive aspect of this approach is its also low encoding and decoding times, which are also an important aspect in our VBR scenario. The encoder was implemented exactly as described in the paper, with only slight adaptations to file input and output.

	ASCII .ply	MVD	MVD*	LDI*	MVLDI*
Size	2.42GB	1.14GB	218.778MB	105.55MB	67.99MB
N. of files	2700	5400	6	80	10

Table 3.4: Video compression results for the Dancer F video dataset. Compressed streams marked with *

Our experiment can be seen in Table 3.4, where we show the obtained results for MVLDI compression, compared to different representation and compression methods; ASCII-based .ply files, which are used in several point visualization applications, raw uncompressed MVD data, compressed MVD and compressed LDI. The dataset used was the "Dancer F", which one single frame is introduced in Table 3.1, a simple wide-baseline capture with three sensors.

MVLDI has the overall best results, while removing more redundant data, and not discarding important data as in an LDI approach (as mentioned in section 3.2). Also, it generates considerably less layers than the LDI approach, creating less total files (10 instead of 80). A high number of generated layers represent a higher number of videos that need to be encoded, and also decoded in real time to be rendered.

While the encoding step can be broken down and performed layer by layer, decompressing and rendering 80 videos in real time is not supported by most available computers, since the allocated memory necessary to perform this task is high, and the process considerably heavy. The ASCII uncompressed .ply file representation shows one of the main drawbacks of a raw point-based representation; a very high number of input files. For each frame a different file has to be read from the disk, which makes the rendering process considerably slower.

When compared to the current approach applied to wide baseline scenarios (MVD), MVLDI occupies 31% of the space, while having comparable video quality, as will be described in Section 4.3.3. It is important to note that this result was obtained with a simple three cameras scenario, where the data redundancy detection showed the lowest advantage (see Section 3.2). The gain on a more complex scenario where data redundancy is higher is expected to be considerably higher.

3.5. Summary

We presented MVLDI, an alternative data representation for a single frame of multiview video that allows wide-baseline VBR applications to take advantage of the redundancy detection of the LDIs. Moreover, MVLDI is also a more efficient alternative to represent a point cloud for an image-based rendering scenario. An alternative redundancy detection technique was introduced, which considers points in a global space opposed to the image-space thresholding of LDI, ensuring a homogeneous sampling of the data.Our results showed that the proposed approach creates a smaller number of layers and detects redundancy at higher rates in both narrow and wide baseline scenarios, while also showing higher efficiency in scenarios with more cameras.

The generated Layers are more dense than the typical residual LDI layers. They can be effectively used for temporal compression on video, and also geometry estimation, as proposed by Merkle et al. (2016).

We also implemented and evaluated four novel approaches to viewpoint selection for MVLDI, and one novel approach for viewpoint generation. Our results showed that search-based approaches are overall more successful, however more research has to be done regarding heuristics for viewpoint selection, and a quasi-complete search algorithm needs to be implemented, opposed to both the greedy approach presented in this paper, and the complete search (*optimal*) which has a very high complexity for video scenarios.

Our results also confirmed that selecting a viewpoint directed according to the normal of the biggest surfaces in the point cloud can be a successful approach, depending highly on the results of the clustering algorithm applied. Specialized techniques for detecting surfaces on highly fragmented point clouds need to be developed, or novel approaches for residual data need to be applied.

Also, a synthetic viewpoint generation approach was presented, being applicable not only in complex wide-baseline VBR scenarios, but specially in visualization of complex point-cloud structures. MVLDI with synthetic view generation is a viable image-based alternative for static point clouds created by sweeping laser scanner captures, having advantages over a traditional LDI.

4
3D Flashback Framework

This chapter will describe the implemented approach for each step of our VBR system, focusing on the contributions to the state of art.

4.1. Overview

Figure 4.1 describes the overall VBR process we implemented to create three-dimensional video content, and highlighting on the red checkered boxed the according steps of the VBR pipeline introduced in section 2.1.2. Some of the described steps have well developed solutions on related literature or software packages, allowing us to focus on the main representation question for this PhD thesis.

The reconstruction and representation steps are the offline section of our pipeline, and are not highly constrained by time. We focus on creating a high-quality data representation that can be used in real time during the visualization stage. A compromise between compression and quality is made. Our goal was to find a representation that can provide this 3D Flashback experience on an ordinary computer,

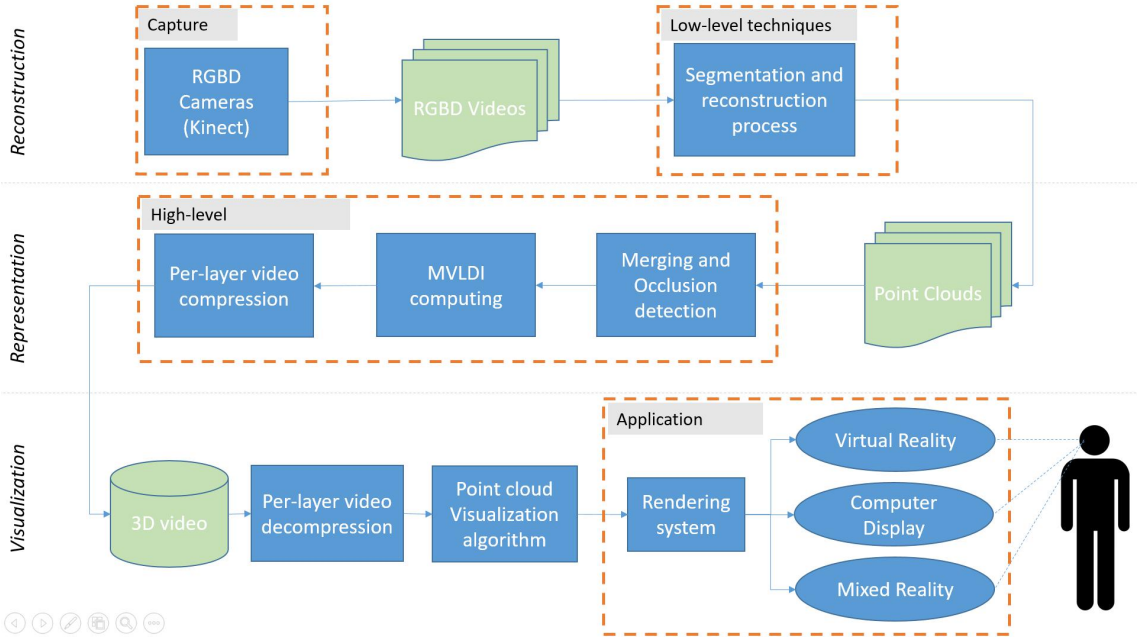


Figure 4.1: Diagram of the Video-based rendering process to obtain three-dimensional videos.

similarly to what is done with video.

Microsoft Kinect devices were used for data capture, since they couple color and depth information as described on section 2.2, and provide us with a better quality for the price of competing sensors. Also, as shown in Figure 4.2 (and also Wasenmüller and Stricker (2017)) the reconstruction technology has improved significantly on the most recent device. Doing so, we produce a set of RGBD videos that can be easily reconstructed into point clouds following the process described on section 4.2.

Segmentation between background and foreground is necessary to deal with the occlusion problem caused by dynamic elements. This can also be important on certain use case scenarios where the focus of the visualization are the foreground elements, such as dance performance, or visualization through mixed reality systems. Also on this step, point clouds are created and filtered in order to have the best reconstruction quality possible before the encoding step, where we can be subject to loss of precision due to data thresholding.

The next step uses higher level techniques based on the point cloud data we produced to create the 3D videos. Initially all the viewpoints are merged and data occlusion can be handled by completing the point cloud with pieces not captured by one sensor. After we finish this second step of processing, we compute the MVLDIs (Section 3.2.2, Anjos et al. (2017)) for each frame, and transform them into MVLDVs by

(a) Kinect 1 male point cloud reconstruction. (b) Kinect 2 male point cloud reconstruction

Figure 4.2: Comparison between 3D reconstruction with the Kinect 1 and 2 devices. The higher detail and smoother reconstruction is still noticed on a zoomed in view.

applying video compression to each layer.

The final reconstructed data will be rendered through a typical rendering system such as OpenGL (low-level rendering API), where the user can freely position the camera and visualize the scene from a freely chosen viewpoint. A specialized point cloud visualization technique was used in order to overcome the point cloud visualization issues (dos Anjos et al., 2017)

Although data is usually displayed on an ordinary computer display, we integrated this system with with Virtual Reality output devices (Oculus Rift) (Section 5.2) providing a more immersive experience of the 3D Flashback application. An alternative output device can be a mixed reality headset, where the user is taken to the same physical place where the data capture was performed, and visualizes only the foreground data being played overlapped the real world as a backdrop.

4.2. Reconstruction

The task of creating a point cloud from a single RGBD camera is a straightforward process. The depth image provided contains at each pixel a distance in millimeters from the camera, and using the pinhole camera model to explain the relationship between these projected values and the original 3D coordinates we are able to reconstruct a point cloud for each captured frame.

The pin-hole camera model describes a perspective projection of a 3D point $P_w = [X\ Y\ Z]^T$ to an image point $[x\ y]^T$ in the image plane. The coordinate system has the optical axis coincident with the Z-axis, which allows stating an equality based in the similarity of triangles $f/Z = x/X = y/Y$, and therefore $x = fX/Z$, $y = fY/Z$,

where f denotes the focal length. Since the origin of the 2D image does not coincide with where the Z-axis intersects the image, we have to add the proper translation to each one of the coordinates, $u = s_x x + o_x$ and $v = s_y y + o_y$, and also that the focal length parameter is not necessarily equal on both axis. Finally one obtains

$$\underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{P_c} \doteq \underbrace{\begin{bmatrix} s_x f & s_\theta f & o_x \\ 0 & s_y f & o_y \\ 0 & 0 & 1 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{P_w} \tag{4.1}$$

where o_x and o_y are the pixel coordinates of the image principal point, $f s_\theta$ is camera skew, which is usually assumed zero in recent cameras, and $\doteq$ indicates equality up to a scale factor. Matrix K denotes the intrinsic parameters matrix.

The intrinsic camera parameters can be found through a camera resectioning process. [1] Several images of a checkerboard pattern were captured in different positions and distances from the camera, the checkerboard corners were detected and tracked on the different images in order to estimate such transformation. Conversion from depth to 3D is then simply achieved by the following relation isolating X and Y on equation 4.1

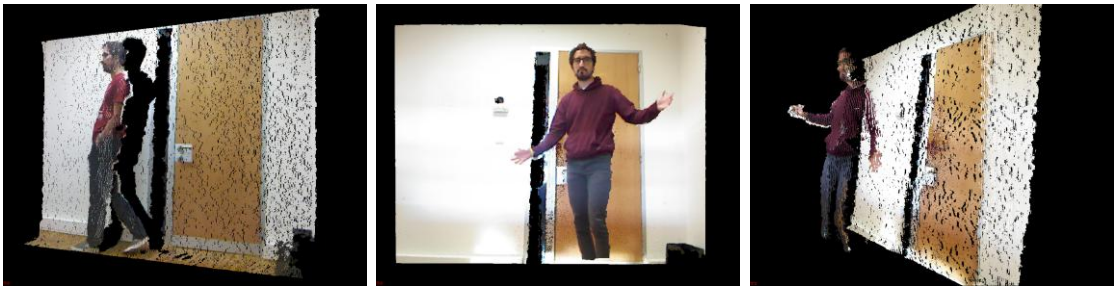
$$\begin{cases} X = Z\frac{(u-o_x)}{s_x f} \\ Y = Z\frac{(v-o_y)}{s_y f} \end{cases} \tag{4.2}$$

This simple calculation produces a series of points (cloud) for each frame, which can also be achieved using the internal SDK functions for the matter, but we chose for the manual calculations for a more efficient conversion.

A point cloud created from a depth image is limited to represent a single depth per pixel. Consequently, occlusions are typically found by even small changes of the viewpoint. Figure 4.3, shows a point cloud obtained from a depth image, and an occlusion created by a person in front of a wall. Although it is known from the other frames that there is a wall on the missing data, the unprocessed cloud of points obtained directly from a depth image will not include that information.

By segmenting the data between foreground and background (Figure 4.2.2) we are able to overcome this issue. This static data is then rendered as a background, and the dynamic elements are played over it. Doing so, we managed to address the oc-

[1] In our system was performed using the OPENCV c++ toolkit.



(a) Missing data due to occlusion. (b) Front view of segmentation approach (c) Recovered wall data from segmentation approach

Figure 4.3: Kinect 1 capture scenarios. First approach where there is missing data on the occluded portion of the wall, and two shots of the solution of reconstructing the missing information from frames where occlusion does not happen, by segmenting the captured data into dynamic and static elements.

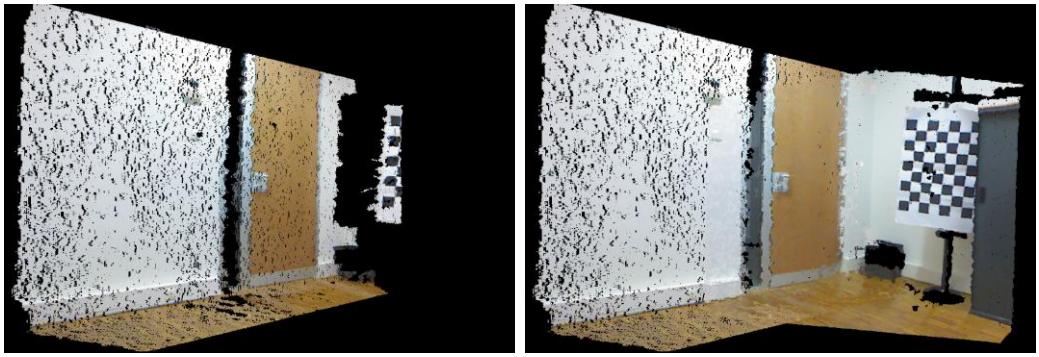
clusion problem by dynamic elements, since we can recover the missing information of the static elements from frames where occlusion does not happen. Results can be seen on Figure 4.3

Regarding the mapping between the color and depth camera, we resort again to camera calibration, in this case the extrinsic calibration between the IR and RGB cameras forming a Kinect. This process can also be performed using the internal SDK functions but with noticeable lower precision. An image point in the IR camera, $[u_i\ v_i]^T$, and the corresponding depth, Z, allow to obtain a 3D point $[X\ Y\ Z\ 1]_i^T$, which can be mapped to the RGB camera using a rigid translation and then projected

$$[u_c\ v_c\ 1]^T \doteq K_c[{}^cR_i\ {}^ct_i][X\ Y\ Z\ 1]_i^T \tag{4.3}$$

where K_c denotes the intrinsic parameters matrix of the RGB camera. Having obtained the coordinates of the image point in the RGB image, then one has the color of the 2D/3D point selected in the IR image, i.e. $I_i(u_i, v_i) = I_c(u_c, v_c)$, where $I(\cdot)$ denotes red, green or blue image components.

In addition to the extrinsic parameters linking the IR and the RGB cameras, one needs also to have the extrinsic parameters connecting different color-depth cameras. In a multi-input scenario, i.e. a scenario composed by multiple color-depth cameras, one has to obtain the rigid transformations connecting the various IR (or RGB) cameras, part of the color-depth cameras. In a multi-input scenario, the rigid transformation, rotation and translation, linking each camera to the reference camera, is applied separately to each one of the output point clouds. The calibration between the different cameras using pattern and feature matching was found to be not very precise due to the resolution limitations of the cameras, and the desired



(a) Missing data due to static element occlusion.

(b) Second kinect stream added, containing the missing data.

Figure 4.4: Information can be occluded from static elements, and the only solution is a multi stream capture that contains that information. The example with two inputs illustrate the issue at hand.

capturing setups we desire to use.

With widely separated cameras precision errors were found to be as high as 24 centimeters. A user-assisted alternative where the user can correct calibration errors has been implemented for calibration on mobile setups (Described in detail in our article Sousa et al. (2017)), and sensors were fixated for our laboratorial experiences.

4.2.1. Data synchronization

As mentioned in section 2.2, synchronization is an essential step for VBR systems. This is an essential question when one has the purpose of combining the data from different inputs into a single output, where errors can be easily verified and detrimental to the desired result. On the context of the distributed virtual environments PhD course, a network-based solution to this problem was implemented.

A very simple yet effective approach is to connect all input devices to a single external trigger that can be started manually, as described in the book from Magnor (2005) about Video-based rendering. However, this approach requires that the input devices accept external triggers, and that one is working with data captured in a controlled environment

Wang et al. (2014b) have developed an interactive solution for synchronization of different streams that match features and pieces of overlapping videos to perform synchronization. The work from Ballan et al. (2010) uses casually captured videos as input, making a synchronized start a non viable option, since the capturing process

is not intentionally started by a single person, but by each individual at a given time. The process described by Hasler et al. (2009) is applied, where the 90% quieter parts of the video are silenced, and the rest is aligned. A rough synchronization is achieved, and some manual adjustments are usually still necessary. Noise synchronization can be a problem when the sound source is far from the capturing position since sound travels slower than light, and the main objective is to synchronize the video track.

On situations where silhouette extraction is possible, Sinha and Pollefeys (2004) developed a technique that uses that information to perform video synchronization for an array of cameras. Tuytelaars and Van Gool on the other hand, use feature detection and tracking to identify five rigid moving objects on each sequence, calculate the trajectories, and match them to the other streams. This approach is preferred on situations where sound based calibration would fail, and also for moving cameras. Similar work from Rao et al. (2003) uses also trajectory estimation for synchronization, providing examples on varied scenarios.

A solution similar to the one from Ahmed and Junejo (2013) was the chosen approach, using networking to synchronize clocks and simulate a physical trigger which starts at a user chosen time point. Differences can be found on the clock synchronization method, which averaged over several queries to a remote server on the case of Ahmed and Junejo, and recording performance. They store the images in memory before sending them to disk in order to guarantee the steady frame-rate that won't be stopped by disk writings, but limiting the amount of captured frames to around 200 pictures.

Given the existing limitations with the Microsoft Kinect sensor being tied to a single computer at each time and hardware input is not possible, software-based networking solution is required. Figure 4.5 shows the physical setup used for this solution. A client and server applications were implemented. Each computer that has a Kinect works as a server accepting requests to either start a capturing process, stop, or capture a single snapshot. A client that is connected to the same network is allowed to message such through the client application that implements the synchronization protocol, and request one of its services. When a server is taking a request from a client, it is blocked from attending other requests in order to ensure recording performance, which would be hindered by extra load on disk writes by different client requests.

Figure 4.6 shows the synchronization protocol implemented. Server machines wait for connections, and when a connection request is given, synchronization between

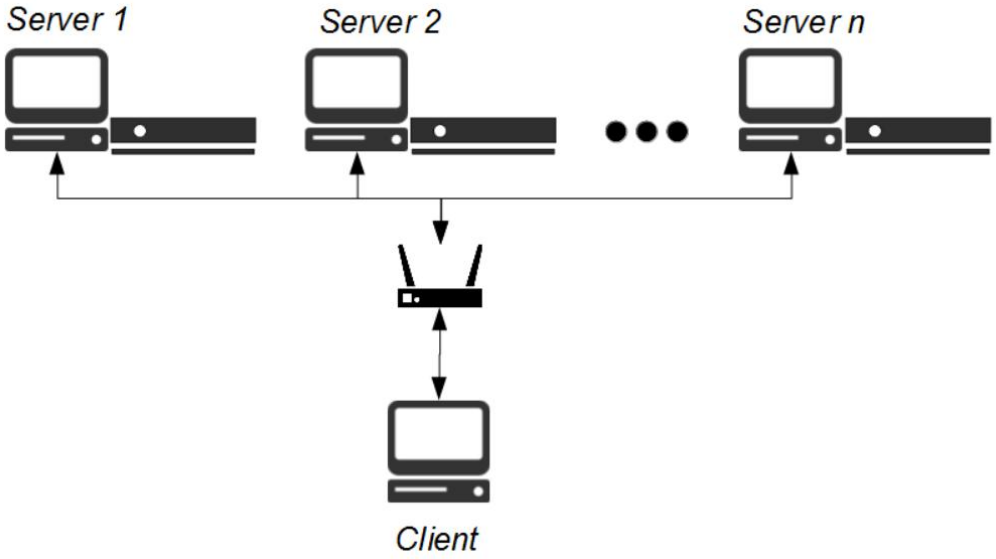


Figure 4.5: Structure of the capturing system

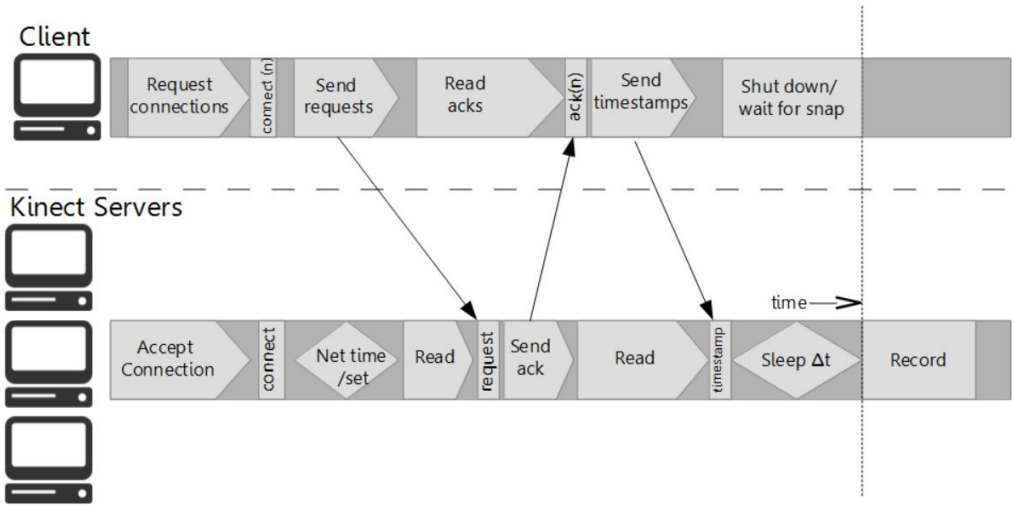


Figure 4.6: Communication protocol for the synchronization process

the clocks of the server machine to the client is performed using the built-in windows net time function, which synchronizes the clocks of two different machines using the Network Time Protocol (NTP) (Mills, 1991). An initial attempt was made with synchronization to a public time server, but when the computers are located on different networks a simplification of the protocol is used which only guarantees a precision within two seconds, which is not useful for our purposes. Ahmed and Junejo (2013) use this approach to perform synchronization, but use several queries until a consensus is achieve about what is the current time. Since we have the objective of performing simultaneous capture on the same physical space, we expect all the computers to be in the same network, being easier to just use direct queries in opposed to a global server.

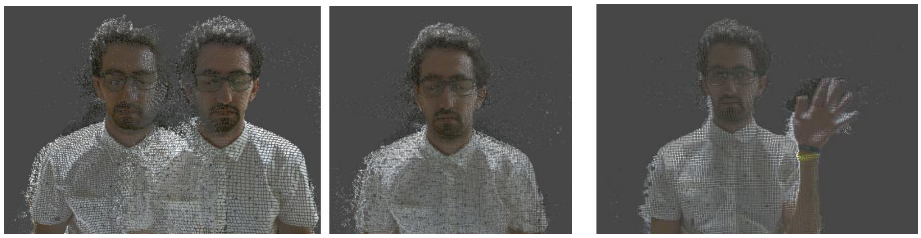
After finishing this step, the server will read the socket for the specific request, which is sent by the client right after connection is established. An acknowledgment message is sent in order for the client not only to know his request was received, but also to know that the time synchronization step has been performed successfully. After all server machines have replied, a time stamp in microseconds of a moment that is a user chosen Δt from the current time is sent. This value can be chosen according to the latency of the network, number of servers, or even related to the nature of the capture performed, such as the client desiring to be present on the capture and needing time to move to the capturing zone.

A message to initiate capture immediately would not be a proper approach given the fact that we would be vulnerable to the latency of the network and message travel times, that is the main reason a delayed approach was chosen, where each server is aware of the time it should start the process, having a synchronized clock with the client.

Each server calculates their own Δt from the moment they receive the message in order to know when to start the capturing process. The sensor is shut off, and the process sleeps until the defined time stamp, when the sensor is turned back on and the recording process is started Video recording is performed locally for better performance due to the high amount of data that otherwise would need to be transferred.

Locally, a steady capturing rate of 30 frames per second is kept. On the case a frame is missed locally due to an unusual delay on the disk writing process, or on the device itself, a blank frame is written and the counter is advanced in order to keep synchronization between different servers. The negative effect of such decision can be minimized through temporal filtering, and interpolating surrounding frames to estimate the missed content. Capture is stopped by a follow-up request through the client, going through exactly the same process, in order to ensure the same number of frames on each point of view. Data is recorded locally, and retrieved by the client through usual means.

The NTP is very successful on synchronizing clocks on a local network being robust to latency (Mills, 1991). The expected error for our system, on a worst case scenario, is close to 0.033 seconds, plus any delays originated from the fact that Windows is not a real time system. This is due to the fact that the Kinect works at a 30 frames per second rate, and might ask for a frame right after one has already disposed or lost by the sensor, making us wait for the following one. Realistically this will most likely not happen since the sensor is started right before capture is initiated, reducing



(a) Blinking; Side by side static frames and combined result.

(b) Fast limb movement while waving with good synchronization.

Figure 4.7: Example scenarios captured to test the synchronization precision. Hard to evaluate on single images but an effort was made to choose frames where movement was being performed.

(a) Idle position

(b) Person walking.

Figure 4.8: One result of a realistic scenario for synchronized VBR capture for three-dimensional videos. Any imprecision was negligible with such sparse data.

our error only to the operating system related factors mentioned above.

Taking the worst case scenario as an example, for a 10 centimeter dislocation to be noticed on the capture, also considering our data is discrete and limited by resolution, the subject would need to be moving at more than 10 kilometers/hour, which is not so likely on the limited space designated for capture.

Figure 4.7 presents two challenging scenarios where a subject is captured blinking and waving his hand, closely to the camera on a parallel setup in order to more precisely see possible differences. The short distance to the camera would precisely capture and represent misalignments. The three-dimensional video was paused at critical times to analyze the quality of the synchronization. Artifacts present are only due to synchronization since camera calibration was performed with static data. The

found results were very satisfactory with close to perfect synchronization between frames when analyzed frame by frame, but perfect when visualized at 30 frames per second. It is important to mention that color information is not precise during movement due to the capturing conditions, and the Microsoft Kinect automatically setting a longer exposure time on its color camera, outputting blurred images. This ends up working in our favor on masking possible errors, and making an argument that a better synchronization would not be needed given the input device being used.

Figure 4.8 shows a realistic scenario where a subject stands idly in front of the sensor and moves naturally on the designed area. Synchronization between different sensors can be seen, not requiring any further adjustments to any of the streams. It was also verified that at this distance where data is sparse, errors on the expected magnitude are negligible and not detrimental to the quality of the final reconstruction.

4.2.2. Data segmentation

As introduced before, foreground and background segmentation is important not only to address certain visualization scenarios such as dance performance, but also to tackle the foreground/background occlusion problem. Although we are able to detect human performers and separate their points in the point cloud to a different dataset, on scenarios where the whole body is not visible, or the user is holding some object, this algorithm will fail to identify such elements.

What addresses both mentioned problems is a background subtraction algorithm, which separates in two different groups of point clouds the dynamic and static elements of a video stream. On the PhD course of Computer Vision, an algorithm was developed to address this particular problem on depth video streams.

As mentioned on Section 2.3.1, there are several different approaches to image segmentation, which are chosen accordingly to the scenario where they are applied. Given the fact that our targeted segmentation classes have to do with movement in the video, we opted for a change detection approach.

Segmentation on Kinect streams has been performed on certain works such as Abramov et al. (2012) and Camplani and Salgado (2014) combines both streams into a 4 dimensional classifier which makes use of the depth information to strengthen the classification. We opted for a depth based approach given the fact that the Kinect 2 sensor has different resolutions for each one of the streams and cannot

properly assign colors to every pixel of the depth image. Edge and change detection on this noisy stream would not give positive results. We based our approach on depth change analysis and used color information for disambiguation.

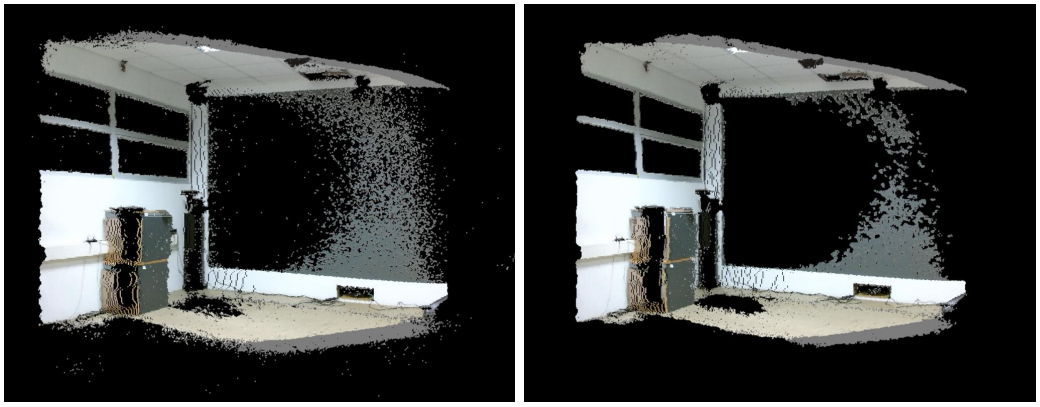
Our approach performs a two pass evaluation on the data, creating a background model on the first run, and properly classifying each point on the second run. Background pixels are estimated on the go, since we wanted to be able to support situations such as opening doors and moving background objects on our interaction scenarios.

Although we have more stable information on the depth streams, there still a big amount of high frequency data on edges or areas with interference of other sensors. Taking a similar approach to the one from Bruhn et al. (2005), our initial step are spatial and temporal smoothing filters to our depth streams. We experimented with Gaussian and Median filters for the spatial transforms, and weighted average for the temporal filters.

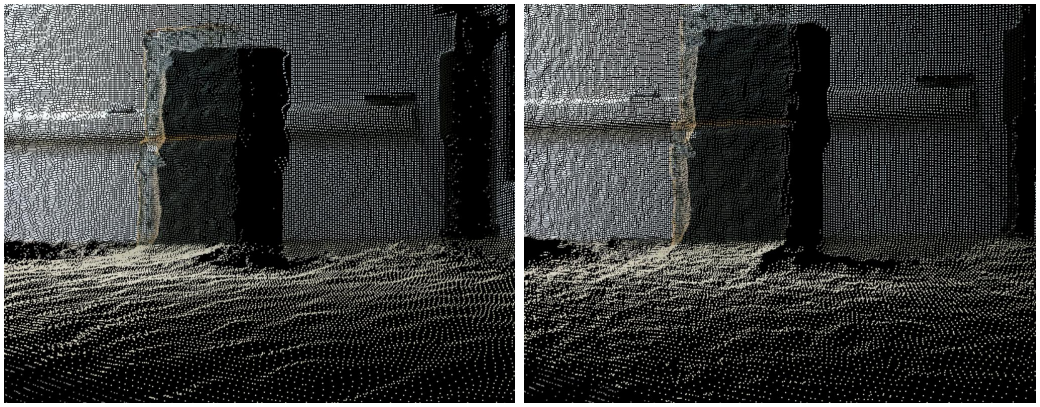
The Gaussian spatial filter showed positive results on reducing the noise, but failed to regularize our surfaces as seen on Figure 4.9c. Weighted averages showed the best results when it came to regularizing the natural oscillation of the depth value provided by the Kinect, but did not address the existence of outliers on a temporal scale. Although this was not a problem on every tested scenario, the results given by the median spatial and temporal filters reduced drastically the amount of outliers found on the segmentation results. Figures 4.9a and 4.9b show the difference found on different filter combinations on the same scene.

After filtering our initial data, the user selects frames where it is known that there is no movement to calculate the average and standard deviation of depth values due simply to noise that remained from the filtering process. Average value will be used as an initial background model for the first pass, and standard deviation as a threshold for segmentation on that pixel.

Our first pass will evaluate the difference between the current frame and our estimated background model, and also assign a tag to each one of the found pixels. If the distance from the background model is smaller than threshold t (defined by the standard deviation or a minimum user defined value) , we keep our background model, updating the color value if the new pixel differs. If the depth difference is bigger than t but not bigger than $2t$, this pixel was previously tagged as background, and the color difference (on the HSV space) is smaller than a threshold c, we consider it a small change, meaning that our previously estimated background model is wrong, since this same color pixel has been slightly moved. We invalidate our



(a) Gaussian and temporal weighted average (b) Spatial and temporal median filters.



(c) Close-up Gaussian filter average (d) Close-up median filter

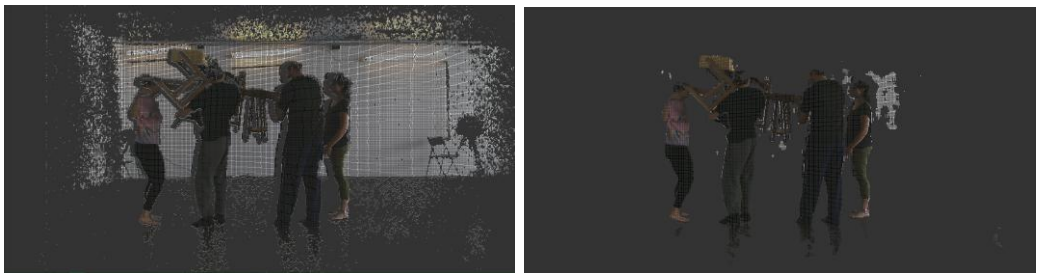
Figure 4.9: Two different combinations of filters on a static environment. Median filters handled better the high frequency outliers. Gaussian filter smooths the surfaces while introducing some irregularities.

background value for that pixel until new information has been uncovered.

If the depth value is found to be bigger than $bg + 2t$, and is located further away from the camera (higher depth value), we consider it to be the end of an occlusion, assigning the new found value to our background model. This value remains until another small change is detected.

After our background estimation is performed, we run the algorithm a second time simply comparing values to the color and depth thresholds. Anything that falls out of our defined thresholds is considered foreground. Color values we found to be very helpful on situations where there is interaction close to walls, but creates new outliers on situations where lighting conditions are affected by the dynamic elements. Fine tuning these parameters to each specific situation is essential for a better output.

Results can be seen on Figure 4.11, where background data from the hallway could



(a) Cloud of a contemporary dance improvisation (b) Result of dynamic elements detection

Figure 4.10: Original and segmented data. Outliers in the background are result of shadows projected by the group of performers.

(a) Background (b) Foreground + background

Figure 4.11: Scenario where door is opened later on, and the system correctly identifies it as a dynamic element, keeping the hallway as the real background.

be estimated when the door was opened, and this element was correctly detected as being dynamic. Shadows produced by the color threshold can also be noticed on Figure 4.10. Our segmentation results fit better our scenario than the built-in Kinect functions, and are flexible enough to be adapted to the capture conditions and elements.

4.3. Visualization

This section will describe the initial experiments performed with existing visualization techniques, and the developed algorithm, and also explain the architecture of the 3D Flashback rendering pipeline. The rendering pipeline was executed using OpenGL or Unity3D (Game Engine) depending on the interaction scenario.

4.3.1. Early results

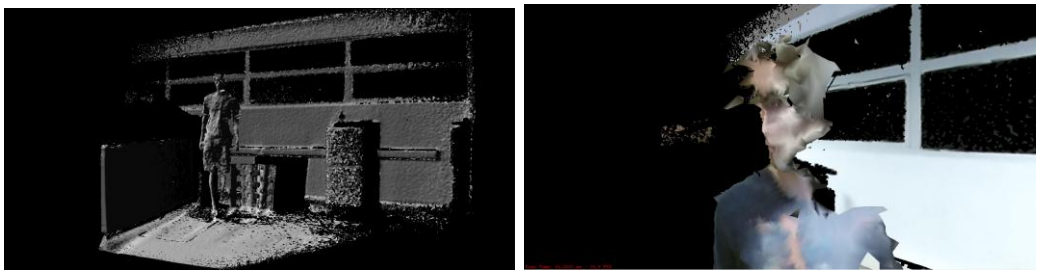
Regarding visualization of each individual frame, when rendering point clouds we have some visualization issues as pointed out by Katz et al. (2007), which greatly hinder the user experience specially when zooming in. The distinction between background and foreground is not clear due to the sparse nature of the reconstructed points, and the absence of element occlusion.

Experiences performed with triangulation algorithms from the Point Cloud Library Rusu and Cousins (2011a) on the point clouds were performed to try to address these issues. Results can be seen on Figure 4.12, where it was noted that on a relatively distant object the quality was far lower than the desired quality for our reconstruction, which solves one of the issues, but creates a new one. Using a different algorithm to estimate the normals Alexa et al. (2003) we achieved a better reconstruction at a high resolution scenario, but not much difference was noted on the usual scenario of a complex scene.

A different approach was tested using a 2.5D adaptation to the Delaunay triangulation [2] algorithm and achieved better results as seen on Figure 4.12d. Textures were easier to apply and results faster and more precise since it is performed using the 2D depth image input. The main limitations to this approach come from the fact that depth cameras typically display noise in depth discontinuities, (e.g. silhouette of a subject) which will generate lots of errors in the triangulation. These artifacts can be seen in Figure 4.12d as white triangles in the silhouette of a subject.

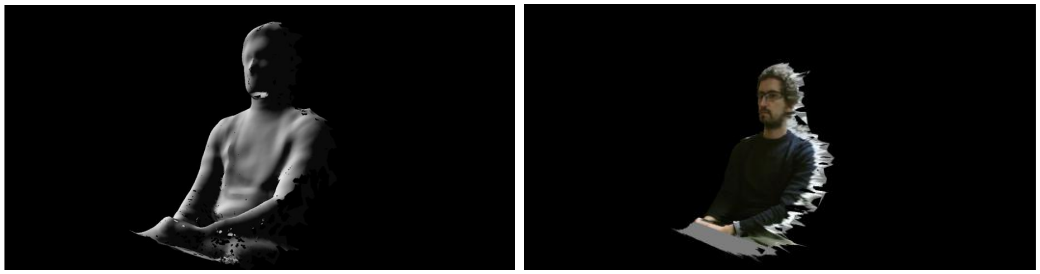
This problem is bigger when a multi-stream capture setup is used, and seams between different captures from different sensors are seen. The noise coming from the discontinuity in each different sensor makes the stitching between different points of view a difficult and time consuming task. Moreover, as seen in the following section, our proposed approach for point cloud visualization has shown results with comparable or superior quality to a mesh reconstruction, while having a considerably smaller computation time.

[2] Fade2D - An easy to use Delaunay Triangulation Library for C++. http://www.geom.at/fade2d/html/



(a) Geometric reconstruction using Moving Least Squares algorithm Alexa et al. (2003)

(b) Triangle reconstruction of a face on low resolution



(c) Close up of higher resolution reconstruction with MLS

(d) 2.5D Delaunay triangulation implementation using depth images as input

Figure 4.12: Comparison between point-based representation and triangle reconstruction. While the foreground/background conflict disappears, smoother surfaces lose quality on a zoomed in view without proper textures. Best result can be seen with the Delaunay triangulation

4.3.2. Stroke-Based Splatting: An Efficient Multi-Resolution Point Cloud Visualization Technique

This section is adapted from the paper published in the "Visual Computer" Journal with the same name.[3]

Point-cloud visualization is a challenging field due to the unstructured nature of the data and its sparsity. Typical mesh reconstructions can be a time consuming task. Splats have shown to have a comparable visual appearance to closed surfaces for visualization uses (Rusinkiewicz and Levoy, 2000; Botsch et al., 2002). However, this is not verified when visualizing a high resolution scan at a given distance, where the size of each splat is bigger than a small region of pixels. This is a common and undesired feature when using low resolution scans or when close-ups are part of the interaction. The shape and colors of each individual splat closely resemble pixelized artifacts from a low resolution image.

[3]dos Anjos et al. (2017)

This section presents stroke-based splatting (SBS), an alternative visualization technique which applies concepts from stroke-based rendering to surface aligned splatting (SAS)). We explore an analogy between art, namely the fin de siécle Impressionist movement, and interactive visualization technology. Since users are culturally trained to interpret this painting metaphor, a higher quality is attributed to point clouds rendered in this painterly fashion, than to a splatted volume, or a blurred mesh on a close up. We conducted a user study that corroborates this claim and also verifies that our approach is perceived as owning a higher quality than state-of-the-art surface aligned elliptical splats (Botsch et al., 2005; Preiner et al., 2012) in all scenarios. A similar approach, the Image-Space Non-Photorealistic Rendering (ISNPR), has been successfully used in the past to visualize scanned volumes (Xu and Chen, 2004) on an architectural context. However, this class of techniques will suffer from aspects inherent to a more flexible 3D interaction or noisier data, as seen in Section 4.3.2.4.

When compared to state-of-the-art point cloud visualization techniques (e.g., surface-aligned elliptical splats), our work contributes to: 1) application of splats using a real world metaphor which allows for richer visualizations; 2) splat shape in the form of brush strokes that allow a clearer perception of the curvature of the object, generated in real-time by a novel lightweight technique; 3) orient each splat more accurately, through the application of the Householder formula, informing the user on the underlying shape of the object; 4) implement an alternative optical splat blending technique which creates smooth surfaces without compromising the interactivity of the system; 5) visualize different size laser-scanned points datasets with less noise; and 6) estimate efficiently tangent vectors leading to higher interactive frame-rates.

4.3.2.1. Point Cloud Visualization Techniques

Surface reconstruction has been the standard approach to visualize point cloud data (Fabio et al., 2003). Several authors have developed successful techniques that estimate the original surfaces from point sets (Gopi et al., 2000; Kazhdan et al., 2006), which are capable of estimating missing data from a faulty reconstruction (Carr et al., 2001) or successfully dealing with noisy input (Kolluri et al., 2004). However, the quality of the reconstruction still lingers on the resolution of the input cloud. Detail on a surface can still be wrongfully interpreted as noise in order to create a smooth output. Given the fact that point clouds naturally require less storage information than triangular meshes, it is still more efficient to use points for

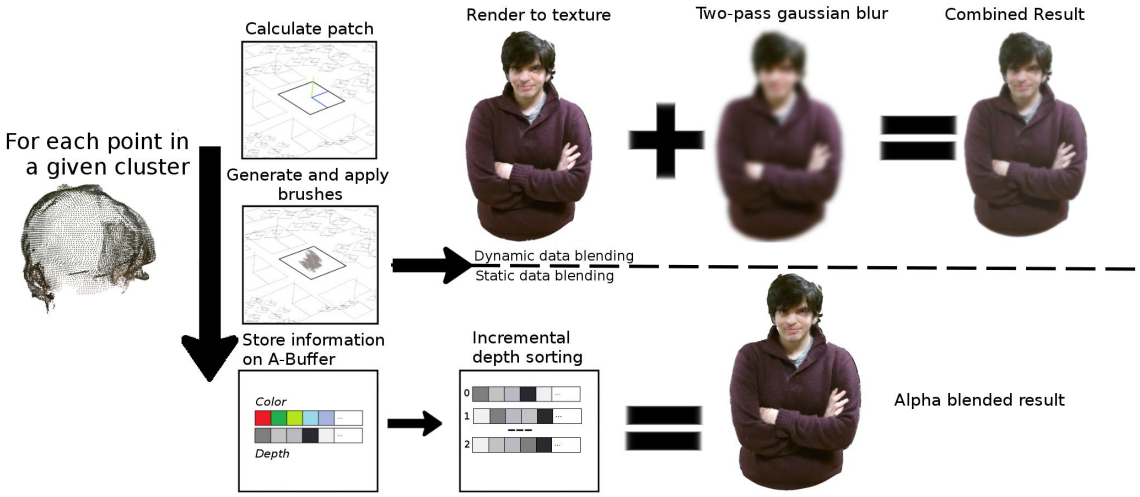


Figure 4.13: Overview of the rendering step

visualization purposes.

Several Point-Based Rendering techniques were proposed that vary according to the primitive chosen to render a point sample as reported by Sainz and Pajarola (2004). Rendering point clouds with point primitives has several drawbacks when compared to other techniques (e.g., background/foreground confusion, loss of definition on close-ups), when confronting a low resolution scenario (Alexa et al., 2003). Katz et al. (2007) solved the problem of background foreground confusion by estimating the direct visibility of sets of points. A similar goal was shared by Awano et al. (2010) for organized point clouds.

Screen-aligned splats (Westover, 1991) have been proposed as a more efficient alternative to polygonal mesh rendering (Rusinkiewicz and Levoy, 2000). More recent solutions combined both approaches (Kawata et al., 2004) to achieve a more efficient rendering of three-dimensional models keeping the quality compromise. Other techniques will align splats to an estimated surface of the object (Preiner et al., 2012; Botsch et al., 2005; Zwicker et al., 2004), which creates a better approximation of the surface. Blending between neighboring patches has been implemented through Gaussian blur (Preiner et al., 2012), alpha normalization (Pajarola et al., 2004), and weighted averages between neighboring patches (Ren et al., 2002).

Nevertheless, the same visual fidelity edges of each individual splat are clearly visible in a close-up interaction, and compromise the quality of the visualized data. Regarding the content or shape of the rendered splats it has not been thoroughly explored in order to improve user experience on close-ups.

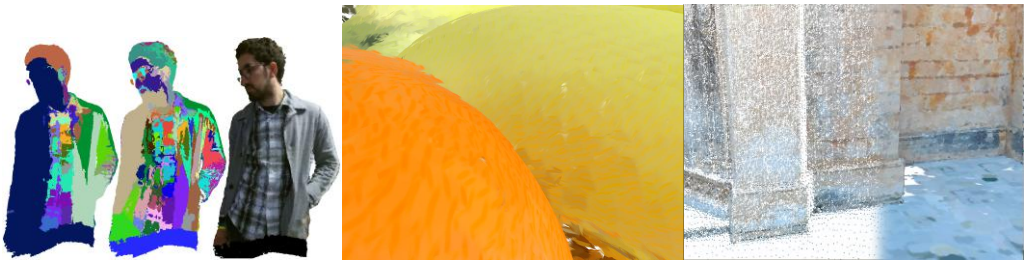
Non-photorealistic rendering (NPR) techniques are normally applied as post pro-

cessing effects on a two-dimensional image space, applying brush strokes at a pixel level (Gooch et al., 2002; Meier, 1996). A wide variety of expressive styles (Curtis et al., 1997; Majumder and Gopi, 2002; Hertzmann, 1998) have been successfully applied to convert images in artistic styles such as Impressionism or Pointillism (Yang and Yang, 2008). The typical NPR scenario aims to process a 2D image to create a stylized output. Several image aspects are analyzed in order to create more adequate brush strokes or textures applied to each part of the image. Using this class of NPR as a tool for point cloud visualization has been explored by Xu and Chen (2004); Xu et al. (2004). Authors also claim that a stylized visualization is sometimes preferred by architects when visualizing buildings, which is also achieved by their NPR technique. Normal vectors and other point information are used to detect feature points instead of image features in a first step, then brush strokes are applied in the rendered 2D image space according to the found 3D features. While this approach might be appropriate to create stylized renderings of images, it has shortcomings when applied to Point cloud visualization (Section 4.3.2.4).

A different paradigm, which is closer to splat rendering and our proposed approach, consists of applying brush strokes or other rendering primitives in a 3D space. Previous work from Runions et al. render point sets as semi-connected ribbons (Runions et al., 2007), which is a non-photorealistic representation that can be viable in some scenarios. The best application of this concept comes from an interactive authoring system where the user manually applies brush strokes to a three-dimensional surface, called Overcoat (Schmid et al., 2011). It enforces the idea that using a 3D brush stroke-based technique to render automatically volumes is a viable and promising alternative visualization providing visually attractive representations of objects. This is the goal of our proposed approach.

4.3.2.2. Description

With laser scanned data, it can be assumed that the points within a close vicinity are equally spaced up to an error value ϵ. By calculating local surface-aligned splats sized $r+\epsilon$, r being the estimated local resolution of points, we can use each splat as a small canvas for our stroke-based technique. These are created according to the estimated normal (Rusu, 2009) together with associated tangent and bitangent vectors (Lopes et al., 2013). Here, we render brush strokes with different properties according to each estimated splat, and apply one of two different blending processes. The resulting "cloud of brush strokes" creates a painting-like visualization of the data. Ultimately, our goal is to provide high visual acuity of every element of the point



(a) Two different clustering settings, and the original model

(b) Small cluster on a group of fruits with two layered painting

(c) Section of a monastery scan with clear distinction between floor and wall resolutions. Splat sizes are correctly estimated

Figure 4.14: Clustering approach and two different application scenarios

cloud at a low cost. However, through manipulating the rendering parameters, the user is able to make different aesthetic choices in order to better fit the application in hand, or simulate a certain painting style.

Our process is divided into two stages: a pre-processing step where parameters are extracted from the input point cloud, and the multi-pass rendering step where the stroke-based splatting technique will be applied (Figure 4.13).

Our technique was implemented using OpenGL and GLSL, version 4.5, on an interactive system that allows for free visualization around the captured point cloud. For the pre-processing stage most of the point cloud manipulation was performed using the publicly available Point Cloud Library (Rusu and Cousins, 2011a).

Pre-processing

A short offline pre-processing step is required to estimate surface parameters that will be used in the rendering step. We aim to maintain brush coherence within each section of the painting, as it is customary in oil painting. This is achieved by taking into account local aspects of the data when defining splat size, orientation, and brush texture.

Clustering

We use a color-based region-growing clustering algorithm as described by Zhan et al. (2009) to group points with positional and color similarities. Clustering parameters can be fine tuned to have a more fine grained segmentation or not. Figure 4.14a shows different clustering settings applied to a real world scanned input. Each splat is sized $r + \epsilon$ where r is the local resolution estimated by the average distance in

the K nearest neighbors of a given point. The error value ϵ can be estimated by the standard deviation of this distance, or set by the user to control the brush size according to the shape of that cluster. Consequentially, denser regions use smaller brushes, while background data makes use of larger brushes. Figure 4.14c shows an example of two different sections of a scan with different resolutions, and the calculated splats according to the necessary size.

The size of the detected cluster is also used to define brush stroke parameters. Small clusters that represent a detailed color region apply a two-layered brush stroke. Although no extra points are added to the data set, the obtained result gives the impression of a higher resolution as seen in Figure 4.14b. Other aspects such as the different proportions on each axis of the cluster or the median color can be used to apply different brush stroke generation parameters to each one of the clusters, in order to have a more faithful representation of a chosen painting style, or simply better suited brush strokes to the data in hand.

Normal estimation

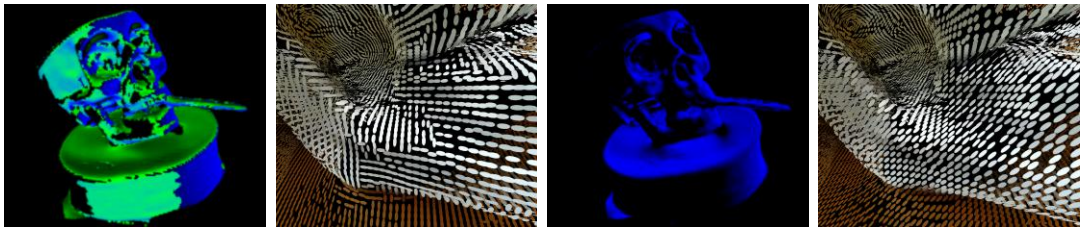
Also in the pre-processing step, normal vectors are estimated by taking the point neighborhood so splats can be aligned to the object at each point. The problem of estimating normals can be solved by finding the least square fitting plane to a local surface S (Shakarji, 1998; Rusu, 2009). We apply this method as implemented by Rusu and Cousins in the Point Cloud Library (Rusu and Cousins, 2011a) to each one of the extracted clusters, searching on an area of radius $3r$, with r being the estimated cluster resolution.

Rendering

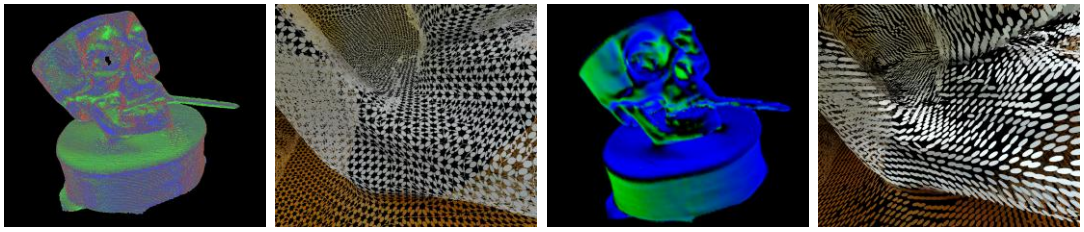
This section describe the tasks performed during the rendering step. Tangential vectors estimation through the Householder formula, brush stroke generation and blending were not calculated in pre-processing time due to the lower impact in the performance when compared to storing the results of an offline calculation.

Tangential vectors calculated with the Householder formula

Similarly to surface aligned splats, we require the determination of an orthogonal reference system composed by normal, tangent and bitangent vectors. Once the normal vector is computed during pre-processing there are several techniques to find orthogonal vectors to the given normal. We chose to use the Householder (HH) formula (Algorithm 7 after Lopes et al. (2013)), which only requires the three components of the normal vector to compute the tangent and bitangent vectors



(a) Eberly (b) Square Plate



(c) Covariance matrix (d) Householder

Figure 4.15: Visual effect of tangential vector computation for the skull dataset. The images reveal the differences between brush point cloud rendering using different tangent vector computation techniques. Left column shows direction of splats, and right column the final visual effect.

to set the splat orientation. Since this formula is deduced from collinearity and orthogonality conditions between the given normal vector and the column vectors of the Householder matrix, the resulting tangent vector formulas are, in most of its domain, continuous and smooth functions that only depend on the normal vector components.

We compared this approach to three other vector orthogonalization techniques in order to explore how well does their brush orientation emulate a natural painting flow features, such as local coherence between brush strokes: 1) the approach described in Eberly (2016) where an orthonormal set is computed based on the cross-product between the given vector and the column of the identity matrix EB; 2) the technique presented in Lopes et al. (2010) where a set of non-collinear vectors is obtained based on the analogy with a square plate (SQP) mechanism and 3) the covariance matrix (CM) technique, which has been the standard approach in surface aligned splatting works (Botsch et al., 2005). Note that HH, EB and SQP only require the three elements of the normal vector at each point to compute the tangent and bitangent vectors (to better access the analyticity of these techniques, refer to Tables A.1, A.2, and A.3 in the Appendix of Lopes et al. (2013)), whereas CM takes the neighboring points as input to define splat orientation. Consequently, the geometric properties of the tangent vectors expressed by HH, EB and SQP are non-shape aware (i.e., do not depend on point coordinates) and extrinsic (i.e., depend on the embedding

Euclidean space) as an isometric transformation such as rotation applied to the point cloud affects the stroke direction. On the other hand, tangents vectors generated by CM produce shape-aware directions which are rotationally invariant. Therefore, in contrast to CM, the HH technique leads to stroke directions not determined by the shape of the point cloud, but by its embedding in space, thus, a rotated version of the point cloud would display differently directed strokes. Nevertheless, this feature can be easily surpassed by computing the HH tangent vectors before rotating the vector field according to the targeted orientation for the point cloud.

Figure 4.15 shows the results for each one of the techniques with the left image indicating the value for the tangent vector, and the corresponding result at the right. The results for EB (Figure 4.15a) show that harsh variations in tangent directions are found in close neighborhoods where similar normal values are to be expected, causing neighboring brush strokes to not follow the expected painting flow. SQP created more continuous tangents (Figure 4.15b), but having a globally similar diagonal direction. These are expected results since EB and SQP do not provide a direct mathematical formula for the desired vector base. Such techniques rather consist of geometric processes involving testing for potential singularities and malformed vectors. Given its analytical nature, the HH formula generates tangential vectors that explicitly depend on the surface normal, thus providing the continuity control necessary to represent locally and globally consistent splat orientations.

Both the CM approach and the HH formula generate rich tangent maps, as illustrated in Figures 4.15c and 4.15d. The disadvantage of using the CM in this scenario is that tangent vectors are not consistent in a close neighborhood, where slight normal vector variations present a larger effect in the calculated tangents. Although mathematically correct, it does not simulate the painting flow we aimed to achieve with this step. We found that the HH formula provided an excellent approximation to this method, while having a higher local continuity. Assuming that the normal vectors are already estimated, EB, SQP, and outstandingly CM are computationally more expensive than the HH technique.

The HH formula calculations are performed on the geometry shader, since it is more effective to calculate tangents in real time than to store three extra vertices for each point in the dataset. Such calculations are not computational heavy, and on large datasets the impact of extra storage would be far more noticeable.

Brush Stroke Generation

Previous stroke-based rendering (SBR) techniques in ISNPR typically resort on the modification of a base brush texture according to image features or user input

Algorithm 7 Pseudo-code for Householder unit vector orthogonalization (after Lopes et al. (2013))

1. Evaluate the sign of the first component, i.e., $sign(n_x)$;
2. Determine the tangent vector with the following expression:

if $n_x \geq 0$ **then**

$$\vec{t} = \begin{bmatrix} -n_y & 1 - \frac{{n_y}^2}{n_x+1} & -\frac{n_y n_z}{n_x+1} \end{bmatrix}^T$$

else

$$\vec{t} = \begin{bmatrix} n_y & 1 + \frac{{n_y}^2}{n_x-1} & -\frac{n_y n_z}{n_x-1} \end{bmatrix}^T$$

end if

3. Determine the binormal vector with the following expression:

if $n_x \geq 0$ **then**

$$\vec{b} = \begin{bmatrix} -n_z & -\frac{n_y n_z}{n_x+1} & 1 - \frac{{n_z}^2}{n_x+1} \end{bmatrix}^T$$

else

$$\vec{b} = \begin{bmatrix} n_z & \frac{n_y n_z}{n_x-1} & 1 + \frac{{n_z}^2}{n_x-1} \end{bmatrix}^T$$

end if

(Haeberli, 1990; Litwinowicz, 1997; Shiraishi and Yamaguchi, 2000; Healey et al., 2004) or fitting to an estimated spline (Hertzmann, 1998). Putting together image analysis with this approach fits to ISNPR, but not to our splatting algorithm as will be discussed in Section 4.3.2.4. Also, image features are computationally heavy to estimate in real-time, and using the same texture for every brush stroke becomes a problem in a close-up inspection (see Section 4.3.2.4)

We define the shape of the brush texture according to a mathematical model based on a combination of Gaussian functions. We generate brush stroke textures with transparency which are applied to each surface aligned splat. The user can configure a set of parameters which are commonly used in SBR techniques, in order to define the base style of brush strokes to be used. Each Gaussian function is represented by the following equation where the resulting z is the aplha value in the RGBA color space:

$$z = Ae^{-\frac{1}{2}(\frac{greenx_\theta - x_0}{\sigma_x})^\gamma} e^{-\frac{1}{2}(\frac{greeny_\theta - y_0}{\sigma_x})^\gamma} \tag{4.4}$$

where,

$$x_\theta = \cos\theta(u - x_0) - \sin\theta(v - y_0) + x_0$$

$$y_\theta = sin\theta(u - x_0) + \cos\theta(v - y_0) + y_0$$

and $P = (u, v)$ the texture coordinates of the fragment being evaluated are the free variables on this equation; A the amplitude of the Gaussian function, which can be used to control the overall opacity of the resulting texture; $P_0 = (x_0, y_0)$ the center

of the Gaussian function on the texture space coordinates; σ_x and σ_y the Gaussian function decay according to the x and y axis; λ is the Gaussian shape exponent which controls the roundness of the function; and θ the orientation of the function in texture space coordinates.

These values (except θ) are controlled by the user and applied globally to the point cloud, but as mentioned in Section 4.3.2.2, these are modified by fixed parameters according to the shape of the detected cluster. An example is adjusting σ_x, σ_y to make longer brushes on longer clusters, or lower γ and similar σ values for rounder brushes.

Variation between neighboring patches is introduced through a noise function with the point coordinates and normals n parameters, applied to the θ value and bounded by user defined values. Figure 4.16 shows an example of how this variation is introduced. From left to right each column of Gaussians share the same θ value, and subsequent columns have a θ that may not differ more than a fixed value from the previous column, in order not to create an unnatural stroking direction.

Multiple Gaussian functions allow us to correctly simulate different effects of a paint brush such as curvature in the painting direction, brush filaments, and uneven concentration of paint. The number and disposition of Gaussian functions is a user chosen parameter. The Gaussians are typically placed in a grid-like disposition, and the user indirectly defines the various P_0 values by controlling the variable dev, which defines the separation between centers. We found that we are already able to simulate widely different brush styles using between 6 to 9 Gaussians on a 3x3 grid, with less having limited expressiveness, and more having diminishing contributions on our scenario. We are also able to simulate two layers of painting by doubling these numbers and generating two overlapped textures with slightly altered color parameters. This was found to have no impact in the performance.

Similarly to splat orientation estimation, this process is performed in realtime, and was implemented using the geometry and fragment shaders. Brush parameters are calculated when processing each point in the geometry shader, and passed to the fragment shader, where the user defined values are also taken into account to paint the current splat with the chosen brush shape, and store information for blending.

Blending Regardless of the user wanting to simulate a certain artistic style or not, blending of neighboring patches was implemented in our system to remove undesired high frequency noise on the rendered images, located on boundaries of patches. These visual artifacts can be seen on ordinary splatting techniques, where the edge

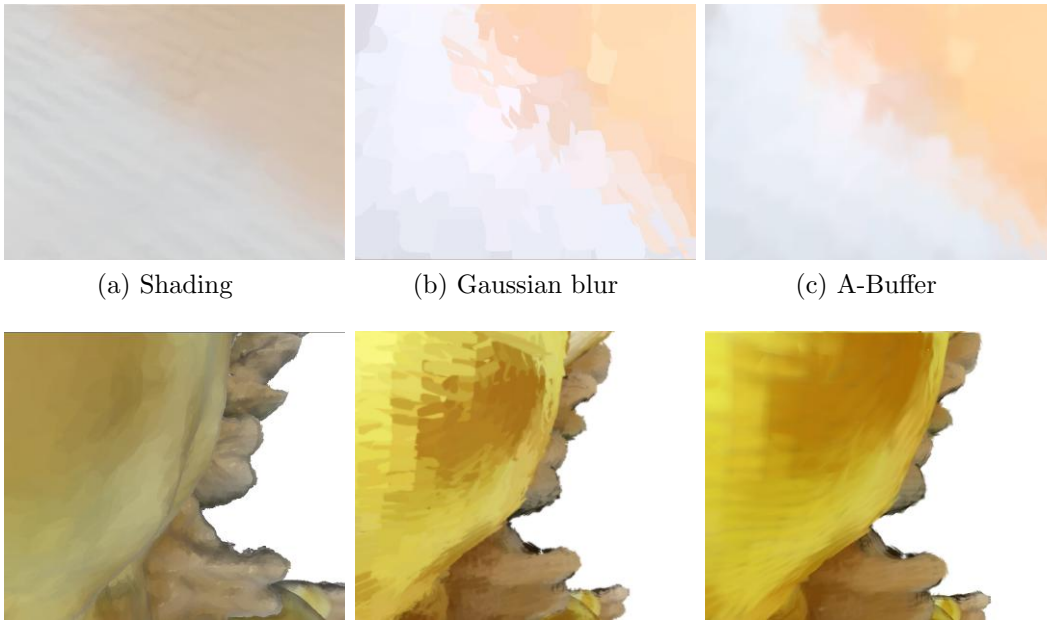
(a) (b) (c) (d)

Figure 4.16: Examples of brush strokes generated automatically using nine Gaussian functions as seen on 4.16a. 4.16b $A = 8$, $dev = 0.03$, $\lambda = 3$, $\sigma_x = 0.114$ and $\sigma_y = 0.0989$,4.16c $A = 1$, $dev = 0.38$, $\lambda = 2.7$, $\sigma_x = 0.142$ and $\sigma_y = 0.071$ and 4.16d with 6 gaussians.$A = 2.27$, $dev = 0.3$, $\lambda = 2.4$, $\sigma_x = 0.0989999$ and $\sigma_y = 0.1259$.

of each patch can be easily noticed. This is something that is not as apparent in the real world, which is perceived as continuous. Previously, Phong shading, (Botsch et al., 2005), global alpha normalization (Pajarola et al., 2004) and local averaging (Ren et al., 2002) were proposed to perform this step. Phong shading will be discussed ahead in Section 4.3.2.4. The other two proposed techniques use the alpha component to perform blending, but do not fit our proposed aesthetic, since averaging will not allow us to perceive the shape of individual splats, thus not revealing the curvature of the objects that is displayed through the individual splats.

As stated on the work from Hertzmann (1998), controlling Gaussian blur allows one to eliminate undesired high frequency noise. The downside of this approach when applied to our stroke based rendering technique, is that blurring has little effect where each individual brush occupies a bigger group of pixels (low-resolution or close-up, Figure 4.17b), which is the scenario where we want to improve visualization quality. Although, when applied to a moving object, or during camera navigation, this technique fills its purpose. A two-pass weighted Gaussian blur was implemented in our system for these two situations.

We implemented an alternative blending technique which relies on the transparency component of the generated brush strokes. This is controlled by adjusting the amplitude, σ_x and σ_y values of the Gaussians. High-frequency noise is removed even with high amplitude values and low sigma values (opaque brush), since the falloff is not instantaneous due to the continuity of the Gaussian function. By applying common alpha blending on the rendered patches (stronger influence for the top-most patches) we are able to closely simulate the process of tint blending. Figure 4.17 shows an example of a boundary region between two colors, and the different



(a) Shading (b) Gaussian blur (c) A-Buffer

(d) Shading 2 (e) Gaussian blur 2 (f) A-Buffer 2

Figure 4.17: Three different blending techniques experimented in the giraffe dataset. A-buffer better simulated the painterly effect, while creating a smooth surface.

techniques applied.

Alpha blending was implemented through the A-Buffer algorithm by using texture memory on the graphics card. Due to the high complexity of this process which would compromise the interactive aspect of the system, we opted for an incremental implementation of the bubble sort algorithm, where its execution is divided across frames. We chose this specific sorting technique since each iteration of this technique orders one new fragment. Each new sorted fragment is used to calculate the alpha blended color, and stored in the last position of the fragment array, simplifying the color calculation operation to a simple mixing of two colors in each subsequent frame. Algorithm 8 describes the technique in a high level pseudo-code.

Algorithm 8 Pseudo-code for A-buffering technique

```
for all pix_uv do
  if all frags sorted then
    c_res = aBuffer[n]
  else
    n_sorted= bubbleSortStep(n_total, n_sortedPrev)
    c = alphaBlendStep(n_total, n_sortedRes, n_sorted)
    aBuffer[n_sortedRes − 1] = c
    c_res = c
  end if
end for
```

The sorted data is only used after a preset amount of iterations, ensuring a minimum amount of fragments to calculate a blended color of neighboring patches. Gaussian blur blending with Z-buffer occlusion is performed in the meantime. Each iteration of bubble-sort operation will enhance the quality of our visualized data, with low impact to the interactivity of the system.

4.3.2.3. User Study

In order to compare our approach with other existing approaches, a user study was conducted with the aim of understanding if SBS had better quality in terms of resolution, surface and shape representation when compared to its alternatives. With this aim in mind, eleven different datasets were used to generate the images representative of each rendering approaches, specifically, mesh reconstruction, surface-aligned splats and our approach. For each technique, different datasets were rendered with different resolutions, shape complexity and color richness.

The study consisted of an online questionnaire composed by eleven 6-point Likert scale questions. Each question had two images, one for either mesh reconstruction or splatting and the other for our approach. The position of each image in each question was randomized in order not to influence/guide the participant in an aesthetic choice. Thirty-one participants filled-out the questionnaire anonymously, completing the required questions. We asked each participant *"to look closely at each image and comparatively analyze which do you think has superior visual quality in terms of resolution, surface and shape representation."*

A summary of the obtained results including descriptive as well as inferential statistics are presented in Table 4.1. Since our sample did not follow a normal distribution, the Wilcoxon signed-rank test was used to assess the statistical significance when comparing the results between approaches. A thorough analysis of the results and their implications of the derived conclusions of this work are presented in sub-section 4.3.2.4.

4.3.2.4. Results and Discussion

Our algorithm was tested on a Intel i7-6700MQ at 3.40GHz desktop computer with 16,0GB and an NVIDIA Tesla K40c paired with a NVIDIA GeForce GTX 980 Ti graphics adapter. The datasets were captured by using the Microsoft Kinect One device (Camplani et al., 2013), with the exception of publicly available point clouds

(a) Point rendering 580k (b) Image-space NPR 580k (c) Our approach 580k

(d) Point rendering 40k (e) Image-space NPR 40k (f) Our approach 40k

Figure 4.18: Two different resolutions for the Monastery dataset under different rendering techniques: 580.062 (top), and 40.363 (bottom). Leftmost pictures contain pure point rendering with a fixed size to show the density of the dataset.

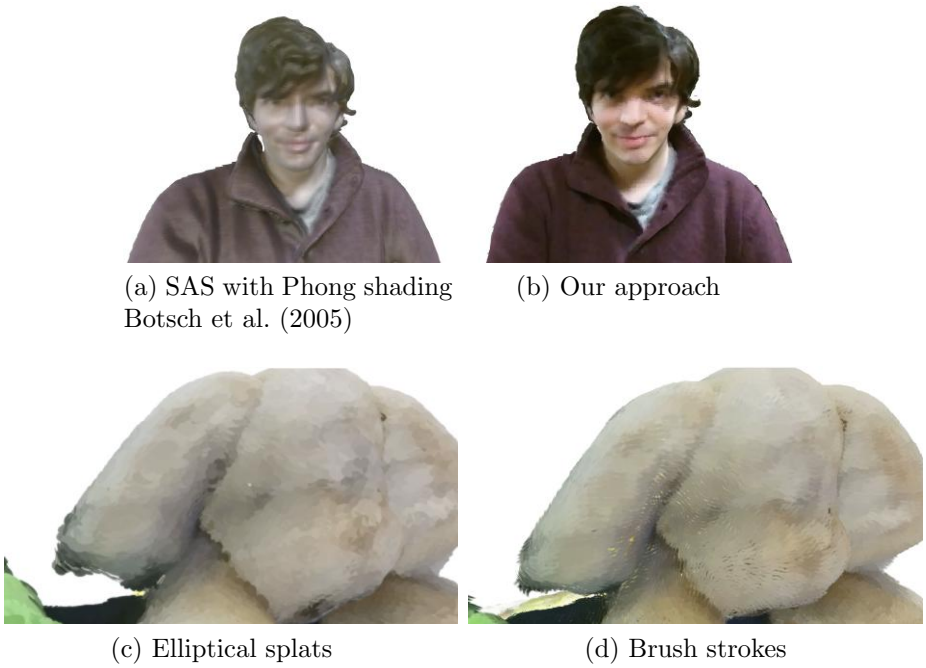
that were used for benchmarking purposes. Although our main objective was to test the viability of our technique on lower resolution point clouds, we tested its scalability on very dense datasets regarding visualization and performance.

Comparison with image-space NPR

We compared our approach with an ISNPR algorithm applied to our Monastery data set at two different resolutions as seen in the top row of Figure 4.18. The most relevant works which use this approach for point cloud visualization are the ones by Xu and Chen (2004); Xu et al. (2004).

In a high resolution dataset viewed at a distance, our rendered result is close to reality green (Figure 4.18c), with just slight differences from the point based rendering. ISNPR will always create a stylized result (Figure 4.18b), regardless of the distance of visualization, which may not be desired in a common point cloud visualization scenario.

When applied to a low resolution dataset (Figure 4.18d), which can be considered the equivalent to zooming into a certain point cloud, point rendering leaves sparse holes, which the big brushes applied by ISNPR techniques are not capable of covering (Figure 4.18e). Our approach creates a closed surface, with only at this lowered resolution (10 % of the original), showing a more stylized visual representation,



(a) SAS with Phong shading Botsch et al. (2005)

(b) Our approach

(c) Elliptical splats

(d) Brush strokes

Figure 4.19: In depth comparison between our technique and Surface Aligned Splats

while maintaining coherence in a close-up interaction.

For the goal of point cloud visualization and solving the problems of background/ foreground resolution and filling holes, our approach is superior to ISNPR, which can only be applied in high resolution datasets, up until a certain distance. For the goal of creating stylized results using point clouds as input, ISNPR has greater range of created effects due to the higher flexibility in the applied brush strokes, but still has the mentioned interaction limitations. Figure 4.18f shows that our approach can create a stylized result similar to an ISNPR approach with only 10 % of the points, and no restrictions to the interaction.

Comparison with elliptical surface aligned splats

When comparing our approach to the state of the art visualization technique for point clouds (i.e., SAS), we acknowledge three main improvements when dealing with laser scanned point clouds. Firstly, the change between elliptical splats to brush strokes. Not only can the impressionistic aesthetic be achieved, which aggregates a higher perceived quality to our results, but curvature can be more clearly perceived than when using elliptical splats. Figures 4.19c and 4.19d show the same dataset with different splat shapes oriented in the same direction. Even from a moderate distance, strokes can be easily perceived, providing better shape perception.

Secondly, color blending through alpha values was more adequate to laser scanned data. This has been previously used through global (Pajarola et al., 2004) and local averaging techniques (Ren et al., 2002), with positive results. These techniques, however, aimed to always create the illusion of a single surface, not revealing the shape and orientation of each single splat, which was one of our goals. Our blending through the A-buffer is not meant to be seen as a replacement to these techniques, but it is a better fit to our painting metaphor.

Current state-of-the-art splatting techniques based on shading (Botsch et al., 2005; Preiner et al., 2012) are more suitable to a scenario where someone wants to replace his mesh rendering pipeline by a splatting technique for an efficiency and quality trade-off. When a mesh is the expected input, one shall have strictly defined materials for each segmented component, and Phong shading can be applied to create high quality representations. With a laser scanner input only color values representing the real world lighting conditions are available. The absence of well defined materials leads to an inadequate aspect for the rendered results, as seen in figures 4.19a and 4.19b.

Finally, splat orientation. CM is considered the standard method to estimate tangent vectors for splat orientation. However, we found that the HH formula presents several advantages when applied to our specific scenario. For instance, HH tangential vectors provide an interesting visual effect where "virtual paint brush strokes"

Dataset	**SBS**	**Mesh**	**SBS**	**Mesh**	**Equal**
Giraffe	3 (2,5)	3 (1)	42%	55%	3%
Yoda	4 (2,5)	4 (2)	48%	39%	13%
Shirt	4 (2)	3 (1)	61%	29%	10%
Man*	5 (2)	5 (1)	13%	42%	45%
	SBS	**Splats**	**SBS**	**Splats**	**Equal**
Frog	4 (1)	4 (2)	68%	32%	0%
Girl*	5 (1,5)	3 (2)	90%	3%	7%
Monastery*	5 (1,5)	4 (3)	71%	23%	6%
Skull*	4 (1,5)	4 (2)	65%	29%	6%
Veggies*	5 (1)	3 (2)	77%	16%	7%
Blonde	4 (2)	4 (1)	52%	26%	23%
Guitar*	4 (2)	3 (3)	45%	16%	39%

Table 4.1: User tests results: Median (Interquartile Range). * indicates statistical significance. Also, percentage of user preference for each rendering technique.

flow smoothly and continuously throughout the point cloud domain, without any unpleasant rendering artifacts. Even though each point is not locally approximated by a quadric surface, the HH tangent vectors do carry first order differential information of the estimated surface normal plane. This contributes to both local and global coherence of the splat orientations.

Despite the richness of geometric attributes associated with the CM technique (i.e., it provides a full reference frame composed by normal, tangent and bi-tangent vectors oriented along along the local and anisotropic density of points), solving the eigenvalue and eigenvector problems are computationally costly as it demands the calculation of the Jacobian matrix and to solve the characteristic polynomial. The number of FLOPS of the HH formula to compute both tangent and bi-tangent vectors, given a unit vector, sums up to: 1 order operation, 3 summations/subtractions, 10-12 multiplications, and 1 division. This is much less demanding than solving the eigenvalue problem for a 3x3 matrix. Hence, the HH technique allows for real-time tangent computation. Note that this advantage holds under the assumption that the normal vectors have been estimated by a more efficient technique than the CM method. For this purpose, there are several approaches to estimate surface normals for point data (Klasing et al., 2009), which must be chosen according to the best graph structure that represents the neighboring points.

Moreover, the HH formula holds for a wider diversity of input data types, namely, depth maps, analytical representations such as implicit surfaces, and meshes or point clouds with high or low quality, while CM is only suitable for good quality meshes and point clouds.

Contrary to the CM techniques which aligns splats along the local and anisotropic density of points, the HH formula uses an isotropic radius as it does not consider point density, only the direction and sense of the local normal vector are used for tangent calculation. This can be seen as advantage whenever the point cloud is locally noisy, under sampled or in the presence of holes, which is the "scenario par excellence" of our work. As illustrated in Figure 4.15c, with the covariance matrix technique, the tangent orientation is less locally coherent than the HH formula, as seen in Figure 4.15d.

Visual perception of point clouds: user test results.

We compared our rendering technique to Meshes and Surface Aligned Splats (Preiner et al., 2012; Botsch et al., 2005), the two most popular visualization techniques to

(a) Shirt: 229.779 points (b) Giraffe: 109.606 points (c) Yoda: 34.878 points

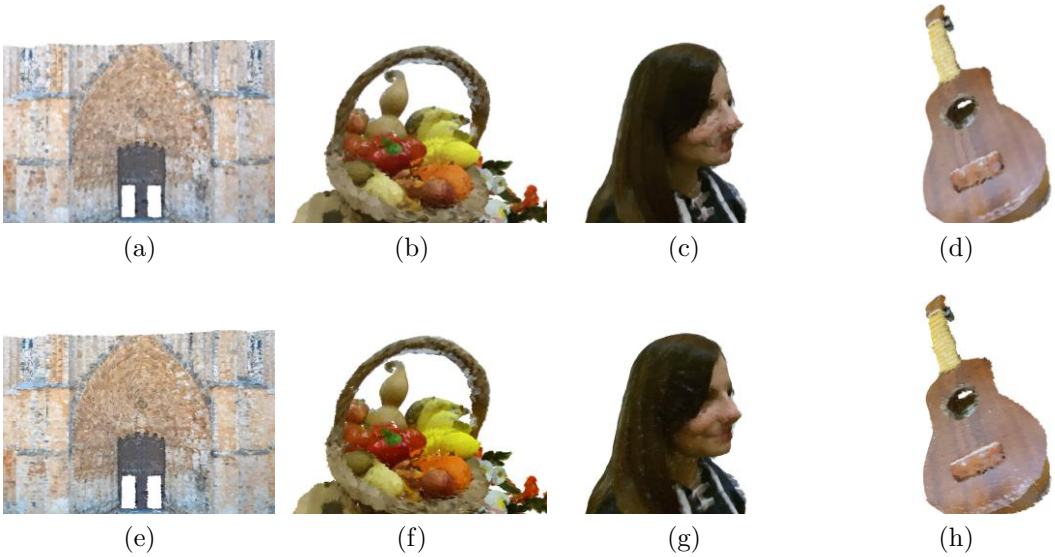
(d) 122.950 points (e) 50.799 points (f) 12.221 points

Figure 4.20: Visual perception results. 4.20a 4.20b and 4.20c: comparison between a unlit mesh rendering (top), and our technique (bottom). SBS has comparable quality to meshes, with a different type of resolution artifacts. While meshes blur details, SBS shows individual strokes. 4.20d 4.20e and 4.20f: Same point cloud with different number of points. On a 10 times smaller cloud, rendering is closer to an artistic depiction, but still allows for accurate visual perception of shapes

represent three-dimensional data. Our results were validated through the presented user study (see Table 4.1).

Figures 4.20a, 4.20b and 4.20c show three different data sets comparing our approach to a mesh. From a distance, similar to splats or point rendering, our technique was perceived as equal to a mesh. This was confirmed for our technique in the Data set "Man" on our user study, where both techniques had the same median value, and the majority (42%) of the test subjects graded both techniques equally. Data sets Giraffe and Yoda (Figures4.20b and 4.20c) which are slightly closer to the captured objects, showed no significant difference between approaches. Both median values and percentage of preference are similar, indicating a similar performance from SBS to meshes.

Although no statistical significance was found on the zoomed in "Shirt" data set



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 4.21: Comparison between our approach (bottom row) and most recent surface oriented splats (Preiner et al., 2012). Biggest improvements noticed specially when fine features should be recognizable, such as 4.21a and 4.21c

results (Table 4.1, and Figure 4.20a), our approach has a higher median value and user preference, with double the amount of users grading SBS higher than a closed in mesh. We believe that these results still indicate the adequacy of SBS on closed in data sets. Meshes display blurred textures due to the loss of resolution when closing up. Our technique does not display blurred data, but individual brush strokes that resemble those in an artistic painting. This can also be noted on Figures 4.20d to 4.20f. A full point cloud, and its 41.3% and 9.9% reduction have comparable visual results. As the number of points decrease, we simply approximate ourselves to a more impressionistic depiction of the rendered model, which is not necessarily perceived as a lower quality one.

Another relevant point comes from the fact that surface reconstruction techniques (Kazhdan et al., 2006; Alexa et al., 2003) lose details on lower resolution point clouds in order to obtain a smooth surface. We argue that in this scenario, our visualization offers a viable alternative to the rendering of a reconstructed surface.

When compared directly to splats, our technique was perceived to have higher quality by the majority of users in every displayed dataset, including the ones with similar median values. Figure 4.21 shows comparisons between surface aligned splats (Preiner et al., 2012) and our technique on the best performing scenarios.

Although less noticeable than in the screen aligned splats approach, they still distort the more delicate shapes of objects (Figure 4.21c, top). On this example where a

familiar shape (human face) is displayed, our approach had the biggest edge over splats, with a difference of 2 in the median quality value, and a 90% preference by the users.

Figure 4.21e shows a highly detailed Monastery entrance where SBS allows for clear perception of individual arches and structures through the orientation of brushes, while circular splats (Figure 4.21a) do not. Figures 4.21b and 4.21d show a zoomed out data set where although no artifacts can be visibly noticed, SBS still was perceived to have a statistically significant advantage over splats. Users consistently identified and referenced our results as paintings in the user study comments, which we argue to be the reason that they were rated with higher quality.

Data set "Skull" had similar median value results between both techniques. This is due to the fact that most of the model was white, and neither individual splats or brushes could be clearly perceived. Similar claims could be made about "Frog" and "Blonde", which have large uniformly colored surfaces. Although similar grades were given, over double the users graded our approach higher than splats in these three scenarios.

Performance

We performed tests using publicly available models from the Stanford scanning repository: Dragon (437.645 points), a data set with three Buddha models combined (1.630.572 points) and Lucy (14.027.872 points). Due to the fact that these models do not have color information, which is necessary for patch size estimation in our technique, we included a laser scan of a Monastery with a similar dimension to the Dragon model. Interaction was always performed on highly interactive frame rates in a 1920x1080 window. Table 4.2 show the performance results for each dataset. Median frames-per-second, first and third quartiles.

The complexity of the rendering activity is directly impacted by the used blending method. When blur is being used, our technique is comparable to pure point cloud rendering. The extra operations performed by our algorithm are due to the calculations of the Householder formula applied to each point to create the local patch, and the combined Gaussian function for the brush strokes at each rendered fragment. All of the necessary operations for these formulas are atomic on the graphics card where they are executed. Due to the use of clustering to determine the size of each brush stroke, the effect of triangle operations such as cutting and z buffering can be considered negligible since we aim for the least amount of overlap between neighbor-

Model (points)	G.Blur(fps)	A-Buffer(fps)
Dragon (566.098)	418 (368:506)	444 (302:642)
3 Buddhas (1.630.572)	254 (247:270)	257 (130:373)
Monastery (580.062)	420 (396:486)	352 (285:505)
Lucy (14.027.872)	67(63:71)	62(56:267)

Table 4.2: Median Frames per second (first : third quartile). Similar median values between both techniques, but higher Inter quartile range on A-buffer.

ing patches. Blurring was found to have a negligible effect on the frame-rate since the two-pass implementation is very efficient in the graphics card, minimizing the amount of texture look-ups through linear filtering.

Alpha blending operations scale with the size of the output screen times the number of layers on the A-buffer (or the limit of operations placed on the bubble sort algorithm for each frame), and the size and shape of the chosen brush strokes. The transparency, shape and size chosen by the user determines the number of fragments at each pixel array during blending. Although these operations have a harsh effect on frame rates, seen by the higher difference between the first and third quartiles, they are only being applied when there is no interaction taking place, and for a short period of time (first steps of the bubble-sort algorithm), hence the low effect on the median frame-rates (Table 4.2). We found that although rendering is process intensive according to the varied mentioned factors, it does not interfere on the system's interactivity.

Given the fact that the visual acuity of a lower resolution point cloud is comparable to a denser one while using our technique, we argue that we can use largely simplified versions of such point clouds without compromising the quality of the interaction. Moreover this allows for rendering and visualizing of larger but sparser data sets, offering a richer interactive experience.

Although the pre-processing step has computational heavy operations such as normal estimation and color-based clustering, which are bound by the KNN algorithm complexity which is $(O(n^{dk+1}logn))$, our goal was not to optimize its execution time. This is justified by the fact that these operations can be performed only once and then stored on a hard disk. Pre-processing times were found to be short on these datasets, with 10 seconds for Dragon, 36 seconds for three buddhas, one minute for Lucy, and 2 minutes 29 seconds for Monastery, due to the higher number of clusters generated. All low resolution datasets had negligible pre-processing times.

Other results

We found that the quality and shape of the brush stroke textures greatly affected the perceived quality of the rendered result. The achieved visual aspect can closely resemble different painting styles through manipulation of the brush stroke textures blending and scale factors. Although our initial focus was not on stylizing, the artistic results achieved, especially on lower resolution data sets, open new possibilities for non photo-realistic rendering techniques (examples in the attached video for this publication). Additionally, the wide variety of brush textures we are able to generate through the combination of Gaussians can be applied in two-dimensional NPR, image processing-related fields or rendering and terrain modeling.

We also applied this technique on time-sequenced point cloud data sets, where we discovered the brush attribution process to be consistent enough to prevent too much visual disturbance. Due to the fact that the varying parameters of the brush generation technique take into account the x and y coordinates of the point in question, the painting seems still on its non-animated components.

4.3.2.5. Limitations

One current limitation to our technique relates to the size of the A-buffer which needs to be allocated for proper blending. On more complex data sets, a lot of memory is inefficiently used to the limit where it might exceed what is available. Out-of core techniques need to be considered on future work.

Secondly, fragments are sorted for each pixel, not for its original patch. If big brushes are being used, neighboring patches can intersect each other in an unnatural way. To stay truthful to our painting metaphor, a different ordering criteria should be applied, which ensures consistency in the order of application of each brush stroke.

Our system is currently targeted at scanned colored point clouds. Non-colored clouds do not work well with our segmentation algorithm, leading to patches with inadequate sizes, as noticed on the Dragon and Buddha models (as seen in the attached video for this publication),. This forces us into using bigger patches in the whole model in order to have closed surfaces, but unnecessarily complicating the A-buffer sorting operation.

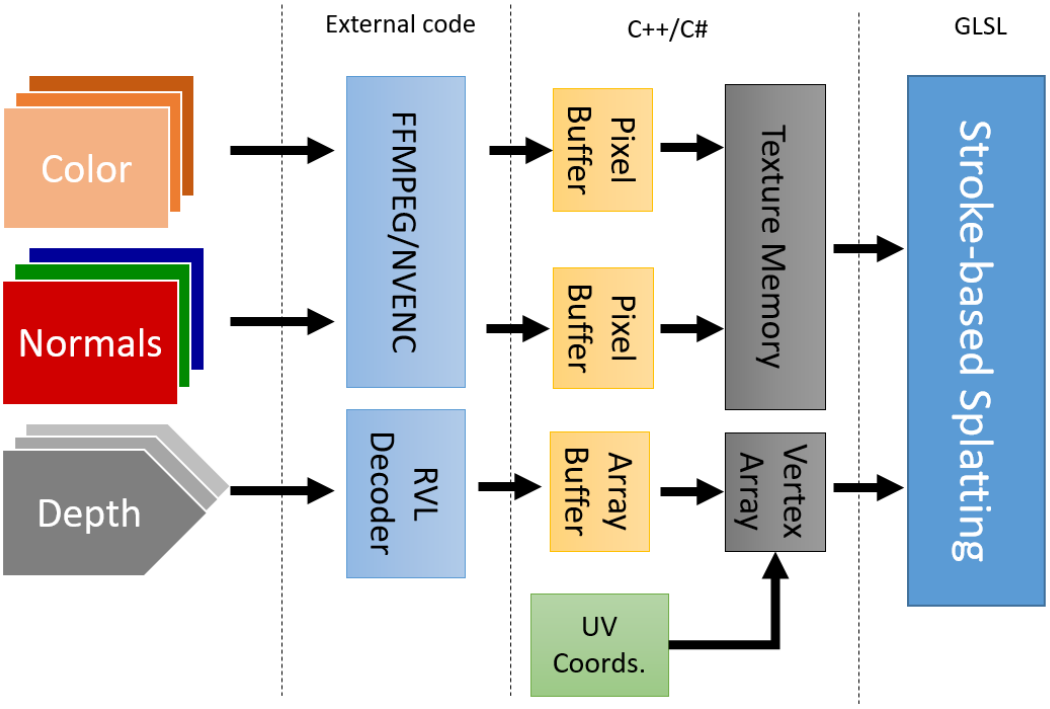


Figure 4.22: Description of the rendering process using the 3D Flashback framework.

4.3.3. MVLDV decompression and rendering

We implemented a C++ based rendering system developed for the proposed 3D Flashback framework. An alternative rendering system in C# was made in order to integrate the framework into Unity3D for easier application development. Figure 4.22 describes the implemented process.

The normal and color streams are decoded normally as video streams (FFPMEG and NVENC were used in our case), updating pixel buffers at 30fps, which will be available in texture memory by the vertex shader. Depth data is decoded using the RVL algorithm (Wilson, 2017), and updated into an array buffer, that is aligned with UV coordinates in a vertex array that is passed to the vertex shader.

The vertex shader uses the process described in Section 4.2 to reconstruct the 3D points from the depth values and the camera intrinsics matrix. Then, each point is multiplied by the extrinsics matrix that represent the viewpoint used to encode that specific frame/layer combination.

Geometry and fragment shaders are implemented as described in Section 4.3.2, using Gaussian blur for color blending instead of alpha-blending. Figure 4.23 shows examples of the obtained output with this dataset.

(a) Dancer F (b) Dancer M (c) Rehearsal

Figure 4.23: Three examples of the implemented system for Multiview Layered Depth Video (MVLDV) decompression and rendering.

4.4. Summary

The presented 3D Flashback framework was fully implemented, introducing effective solutions for the reconstruction step, and novel approaches to both the representation and visualization steps.

A network based method for a Kinect-based VBR system was presented with precision that was adequate to the capturing scenarios. A simple protocol that simulates a delayed physical trigger was implemented with the estimated error being tied to the used operating system not working in real time, and factors that were proven not to be detrimental to the captured situations. For scenarios where faster activities such as sports, more tests will need to be performed, but results enforce the confidence that a better result with the same input device, and the same type of output, is very hard to achieve.

An efficient technique for RGBD data segmentation was presented for a Video-based rendering scenario where the foreground data is the focus of interaction. By iteratively constructing a background model we are able to correctly classify moving elements on the background as dynamic elements, allowing interactions with doors and big objects to have the expected results during visualization.

Combining color and depth values we are able to correctly identify moving objects near the background elements, and also elements with similar color to the background but dynamic during the capture. Improvements still have to be made on noise removal due to lighting variations and shadows projected on the walls. Also the movement threshold values can be improved to lessen the weight of the user chosen values.

We also introduced a novel rendering technique that is better suited to nowadays typical point cloud capture and visualization scenarios. We are able to have equal

or better visual quality to the alternatives when rendering point clouds with lower point resolution, while allowing a less restrictive interaction. Stroke-based splatting was shown to have comparable results when using sub-sampled versions of the point clouds, which indicates its viability for visualizing large data sets.

Compared to splat rendering, our technique was always perceived as having a higher visual quality. We improved the direction, content, and color of each individual splat, while addressing specific problems of point clouds scanned with commodity depth sensors. When compared to meshes, we provide a stylized rendering at lower resolutions and close-ups instead of the blurred visualization typical to meshes, while maintaining visual quality at a higher point count.

A technical contribution of this technique consists of applying Householder formula to calculate both locally and globally coherent tangential vectors that are used to orient brush strokes in space. Also, a simple brush stroke generation formula was presented, allowing for generation of widely varying textures that can be used on non photo-realistic rendering techniques, image editing applications, terrain modeling, and others.

On the field of NPR, our technique is a better alternative to generating stylized visualizations in an interactive scenario, while still being capable of simulating a wide array of styles, adjusted in real time. Overall, The presented framework can be used to render and visualize time sequenced point clouds in real time at interactive rates.

5 Applications

The 3D Flashback framework was used to create different interaction scenarios in the context of the BlackBox Project. Our focus for the initial case studies was to analyze the creative process of the choreographers, and create tools that would support this process. The first presented application is a 2D-3D annotator (Section 5.1 Ribeiro et al. (2016)), which extends an already existing system of video annotation, using the 3D Flashback framework to create and render 3D videos instead. Secondly, we changed the interaction paradigm of the previous application to Virtual Reality 5.2, allowing new types of annotation to be introduced. Here, 3D flashback was also used to create and render the videos. Lastly, we present a concrete case study of analysis on a creative process, where the 3D Flashback framework is intertwined with other processes applied to the point clouds.

All of the described applications in this chapter have used the fully described 3D Flashback framework, except for the MVLDI representation that was under development during the publishing of such articles. Since its completion, such applications have adopted the developed representation. Each of the following sections represents one applied scenario that was published in scientific conferences.

5.1. 2D - 3D video annotator

This section is adapted from a full paper published at the International Workshop on Movement and Computing (MOCO) in 2016 [1]

The global performing arts community has the need for innovative systems which: document, transmit and preserve the unexplored knowledge contained in performance composition processes; and assist artists with tools to facilitate their choreographic or dramaturgic practices, preferably on a collaborative basis.

At present, existing digital archives for performing arts mostly function as linear e-libraries, not allowing higher degrees of interactivity or active user intervention. One of the reasons for this limitation is related to the reduced abilities offered by available video annotators. Specifically, current video annotators support a very limited set of 2D annotation types such as text, audio, marks, hyperlinks and pen annotations. Some of them offer animation functions, which are only applied to ballet choreographies where numerous notation systems exist that allow to represent a very high number of movements and combinations of movements (Moghaddam et al., 2014). This is not the case for contemporary dance, where the movement is unpredictable and can change with every execution either during rehearsals or in live performances.

Previously to the BlackBox project, the Creation-Tool video annotator was developed to facilitate choreographers in analyzing and improving their work, by recording and annotating a rehearsal or a live performance for later review or for sharing their notes with the performers (Cabral et al., 2012). Although this tool has provided significant advances, it still presents the limitations previously described. The BlackBox project endeavours to fill this gap and create a new paradigm for the documentation of performance composition by augmenting a 2D video annotator with 3D visualizations of the resulting annotations. This section describes a system that couples a 2D multi-view performance captured with video and a 3D multi-view captured with Kinect sensors. The paper was entitled "3D Annotation in Contemporary Dance: Enhancing the Creation-Tool Video Annotator". This system translates the 2D annotations taken on the Creation-Tool to the captured point clouds by using feature matching and image segmentation. These annotations can be later visualized in a moving point cloud using an arbitrary viewpoint. The main contributions were the following:

[1] Ribeiro et al. (2016)

- a novel system that combines Computer Vision and Visualization techniques to enhance video annotations with a 3D representation; and
- its application to a live contemporary dance improvisation, demonstrating the possibilities of exploring this novel form as a means to document and preserve the implicit knowledge contained in performance composition processes.

5.1.1. Related Work

Traditionally, performing arts such as dance are taught either by example or by looking at conventional scores on paper. With different dance movements emerging and the impossibility of creating a controlled vocabulary of movements in order to compose a score, watching videos of previous performances or of rehearsals is often the way to learn a specific piece. A common video, though, is not sufficient to communicate what is envisioned by the choreographer (Guest, 1984).

Video annotation systems have been used in this field in order to shorten the knowledge gap between the choreographer and dancers. Non-specific systems such as ELAN (Wittenburg et al., 2006) or Anvil (Kipp, 2010) can be used, or software targeted for this purpose such as Dance Designer (ChoreoPro, 2014), ReEnact (James et al., 2014), Danceforms (Credo Interactive, 2014), Motion Bank Piecemaker2GO (PM2GO)[2] and the Sketch-Based Dance Choreography developed by Moghaddam et al. (2014).

Dance Designer (ChoreoPro, 2014) allows a choreographer to automatically synchronize several aspects of a dance performance, namely the path dancers took, dancers' formation, dance counts, notes and video which makes a complete choreographic score. DanceForms (Credo Interactive, 2014) uses a different approach instead 2-D video. In particular it contains a set of dance poses and corresponding animations that can be combined to create a digital choreography. Moreover, the software also includes common dance sequences, schematic information about dance poses and also basic compositions of individual poses. Although this software includes 3D, and several interesting features, the avatars and resulting poses are not realistic enough, which might be a limitation when using this software to capture nuances and small details that can characterize a choreographer's work.

In PM2GO and Dance Designer ChoreoPro (2014), cue and annotation information appears in its corresponding window and never as an overlay of the video. The

[2]http://motionbank.org/en

annotation type supported is text and can be colour-coded according to a user-defined criteria.

Moghaddam et al. (2014) have developed a sketch-based system that allows the user to compose a digital choreography by individually sketching each individual dance pose. The poses can then be combined and re-arranged in a user-specified order. Each pose corresponds to an animation applied to a 3D avatar shown in two split screens, one showing a god view and the other a third person view. The user can also sketch the path the avatar follows when performing the digital choreography. In contrast with DanceForms, this system does not support any form of annotations or score information.

James et al. (2014) have also developed a sketch-based system that allows composing a digital choreography. There are two main differences between these systems. Firstly, dance poses are presented by video sketches retrieved from the UK Digital Dance Archives. Secondly, the resulting choreography is a motion graph composed of these video sketches, which the user can then fine tune on a key frame basis to achieve the desired final choreography.

One problem they share is the fact that they are limited to a single point of view, introducing ambiguity in the case of occlusions caused by other dancers or props included in the performance. The compromise that has to be made when having 3D representations of the movements is whether to define a subset of movements to be displayed on a virtual avatar, or to have an unnatural visualization of the result with sketches or text annotation on synthetic representations of the performers, which strays away from the traditional experience of watching a performance in order to learn dance.

5.1.2. Augmenting the Creation-Tool with 3D annotations

Our work tries to combine the best of both existing approaches. We argue that the best alternative is to have a full three-dimensional reconstruction of the performance where the user can watch it as a free viewpoint video, with merged annotations from each one of the recording streams. Capturing three-dimensional data using depth sensors of a live performance and enhancing it with annotations is our solution for contemporary dance choreographers to document, analyze, and share their compositions (see Fig. 5.1).

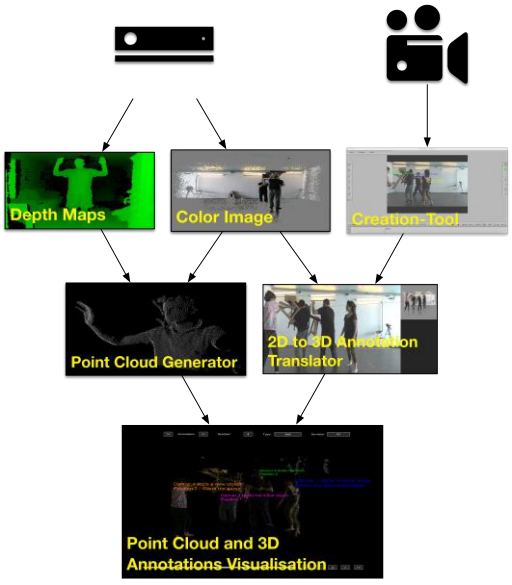


Figure 5.1: Schematic representation of the developed system. At the top, the data input devices: Microsoft Kinect and video camera. In the middle, two modules used to generate the point clouds (left) and convert 2D annotations into 3D ones (right). At the bottom, the 3D Annotation Visualizer built over Unity3D.

We use a wide-baseline setup where each different viewpoint is captured by both by a colour camera and a Kinect sensor. The Kinect sensor is coupled in close proximity with the video camera in order not to introduce a large stereo disparity on the annotation conversion process. Each viewpoint is positioned in opposite sides of the room in order to minimize loss of data due to occlusions. Initially a program was developed which captured the required information using only the Kinect. Nevertheless, initial tests showed that capturing depth maps, colour images and video simultaneously using solely the Kinect resulted in high frame losses, compromising the quality of the 3-D data visualization. In order to solve this limitation, a video camera was attached and aligned to each Kinect used in the final setup (see Fig. 5.2).

Depth streams are synchronized according to the description available in Section 4.2.1 Frame-level synchronization between colour cameras and depth streams is not essential due to the fact that it will only be used to set the starting time of the annotations. Therefore, synchronization based on visual cues is enough for this purpose. Calibration was performed with Creepy Tracker (Sousa et al., 2017), and OpenCV (Itseez, 2015)

Annotations are made on each one of the 2D videos using the Creation-Tool, which encodes them on an XML file with the 2-D pixel position and starting times. Another

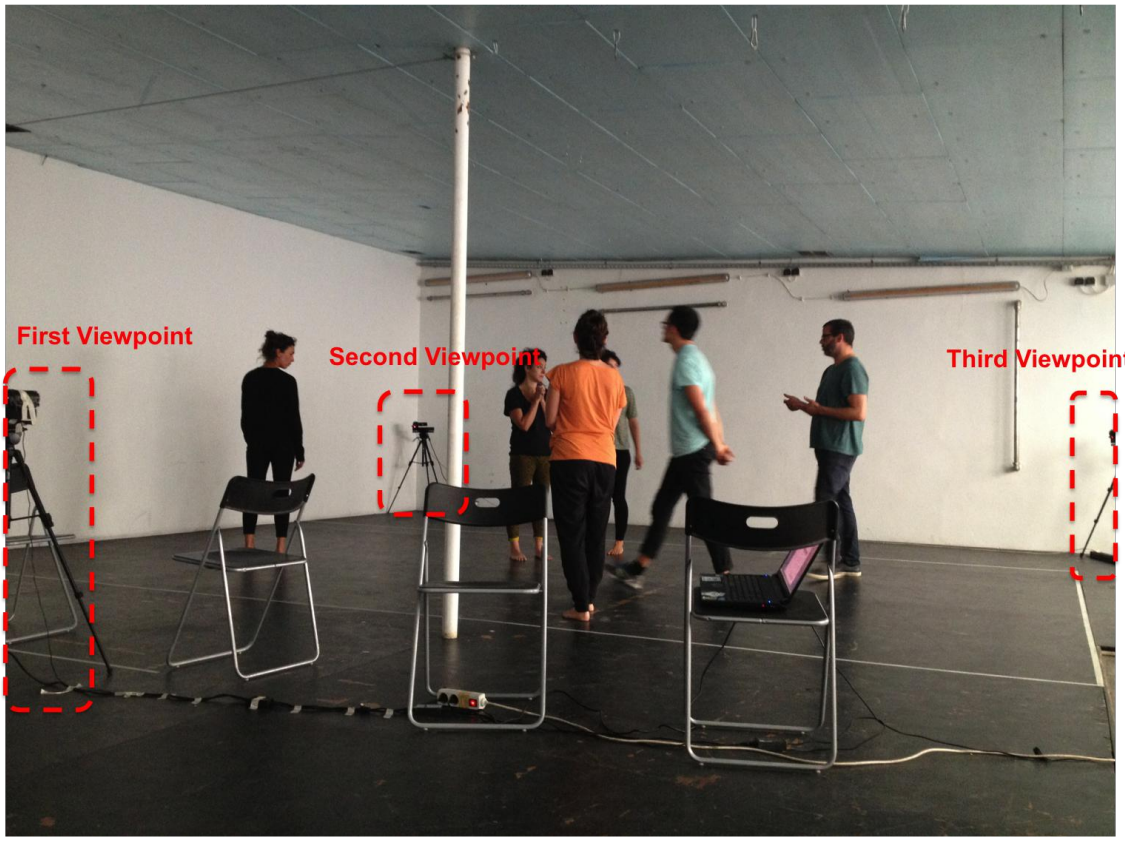


Figure 5.2: Representation of the setup used with three Kinects and video cameras during the case-study done with contemporary choreographer João Fiadeiro and a group of his dancers.

external developed tool to process the XML file and assign 3D coordinates to the 2-D annotations so that they can be visualized in the point cloud video. This tool uses SURF feature detectors (Bay et al., 2008) on the video frame and on the colour frame from the Kinect sensor. The closest feature to the annotation position is selected, and its corresponding match on the Kinect frame is used for depth information in order to calculate the three-dimensional coordinates.

The low resolution and high noise frequency of the Kinect streams reduce the quality of the matches to a lower level than expected. We found that performing background subtraction greatly improves the quality of the feature matching process, with the downside of losing the capability of annotating the static elements of the scene. In the context of our case study, this is not problematic, since our main focus is on annotating dancers in movement.

Point clouds are generated using the depth information (Section 2.5.1.1, and all the streams are integrated in a single point cloud based on the calibration data (position and rotation) of each viewpoint. Since the MVLDI was not completely developed

by the time of this publication, so only short moments of each performance at a time could be visualized, but have now integrated these developments having no time restriction on the visualized data. We have used Unity3D as a platform for rendering the recorded datasets, allowing the user to navigate the camera freely around the performance, as well as to control the playback of the video segment. Annotations are displayed at the three-dimensional position that best matches the point clicked on the original video.

5.1.3. Case Study - Real Time Composition

Within the context of the BlackBox project, we have been working with the Portuguese choreographer João Fiadeiro, who has done extensive work to developed a method for improvisation in contemporary dance, the Composição em Tempo Real (CTR) method (Fiadeiro, 2007). To understand and analyse his method, two separate sessions of Composição em Tempo Real (CTR) involving João Fiadeiro and seven dancers from his company were recorded using the setup described above.

Three different point of views were captured during the improvisation sessions. The space available for the improvisation was delimited on the floor with duct tape to prevent the dancers from getting out of the space captured by the cameras and the Kinects. Each improvisation session lasted around three hours including breaks and feedback discussions related to what was being composed by the group during each set (or improvisation sub-session).

The collected data was used to generate the corresponding point clouds, which were annotated using the respective videos in the Creation-Tool. Audio, text and hyperlink annotations, were supported in the 3D transposition. Ink annotations are not easily translated from 2D to 3D, and will be discussed in Section 5.2

The results can bee seen in Figures 5.3c 5.3d and 5.3e, which present the same scene from the three different viewpoints captured with the three video-Kinect pairs.

For link annotations, the visualization is similar to that of text annotations with the added possibility of opening the respective web page in a browser within Unity3D (see Figure representing an added bonus to this system.

This 3D environment provides a richer setting to analyze and understand contemporary dance, since it allows to observe a particular scene from multiple points of view and therefore solving the problems of occlusion in 2D video. Moreover,

(a) First example of the Creation-Tool: text annotation regarding the RTC method.

(b) Visualizer interface

(c) (Viewpoint 1)

(d) (Viewpoint 2)

(e) (Viewpoint 3)

Figure 5.3: Result of the 3D Annotator Visualizer for the annotations depicted in Figure 5.3a

this representation presents less ambiguity for future researchers studying a particular performance or dancer on their own, since it preserves the correct locations of the annotations. And as it allows to capture different points of view, it can potentially be extended to support collaboration as well, either in real-time or in post-processing.

5.2. Virtual Reality Annotator

This work section is based on the work presented at the ITN-DCH 2017, and published as a book chapter in the Lecture Notes on Computer Science Book Series, "Digital Cultural Heritage".

The previous Section introduced our work with further developments in the field of video annotation. However, annotating in a 2D environment and transposing the data to 3D presents several limitations. The mapping between views is not a straightforward task, and the 2D input during annotation limits the possibilities for other type of annotations that can make use of 3D annotations.

To overcome the limitations of transposing 2D annotations to a 3D environment,

we have developed the Virtual Reality Annotator that allows users to annotate specific body parts and movement sequences in a three-dimensional virtual reality space. The system was implemented in Unity3D[3] integrated with the Oculus Rift V2 development kit, and the interaction with the point cloud and skeleton data is provided by a wireless mouse.

We start by describing background work related to video annotators and their uses in the context of dance, following with the Virtual Reality Annotator where we describe data capture and visualization, the software architecture and interaction, and a discussion the obtained results.

5.2.1. Related Work

This section only will review related work for Virtual and Augmented reality in the area of dance performance, and general annotation and visualization through virtual reality (VR) and augmented reality (AR) systems. For more related work on regular video annotation, please read Section 5.1.1.

Wearable technology and motion tracking have developed to the point where VR and AR are usable in a dance context with minimal disruption to its traditional practice. The article from Gould (2014) discusses "AR art and performance", and how the mixture with technology creates a different type of art, putting the *"body at the heart of its becoming"*. The displayed content can now depend heavily on the perspective, gestures, and positions of the body of the one whos visualizing the work. One given example is the "Eyebeam" project where a dancer is partnered with an AR avatar which can only be seen through a device. A similar goal is shared by the WhoLoDance (Camurri et al., 2016), which already uses head mounted display (HMD) technology.

Both approaches show the importance of embodied systems and presence Sanchez-Vives and Slater (2005) in the context of dance. This has been used as a different approach for teaching dance. Kyan et al. (2015) developed a system to train ballet poses in a cave environment, similarly to previous work from Chan et al. (2011). However, by not displaying the virtual world through an HMD, the sense of presence and body ownership is considerably lower.

Annotating through an HMD has been mainly targeted at AR scenarios, where real world problems can be observed and tagged for later inspection. The survey pa-

[3]https://unity3d.com

per from Wither et al. (2009) reviews these types of annotations in great depth. Virtual reality has not been thoroughly used for video annotation, due to the fact that free-viewpoint videos and point cloud based data that register real world events still not being commonplace. Different techniques have been proposed to annotate static point clouds in an non immersive scenario (Veit and Capobianco, 2014; Bacim et al., 2014), but have limitations when translating them to HMD, where one cannot resort to using hand-held devices with an auxiliary screen, or other peripherals such as keyboard for input. Static inspection (Addison and Gaiani, 2000) and annotation of point clouds (Lubos et al., 2014) has been done using VR, with a focus on architectural problems and rich environments. Using the advantages of embodied experiences in dance to annotate captured point cloud videos through a HMD is a problem that has yet to be addressed.

5.2.2. Description

Data was captured using the described 3D Flashback framework in Section 2.5.1.1, associated with a skeleton recording module using Creepy Tracker (Sousa et al., 2017) We used Unity3D as a platform for rendering the recorded datasets, where the user could freely navigate the camera around the performance scene.

5.2.2.1. Architecture

The Virtual Reality Annotator is a network-based application (see Fig. 5.4) that integrates several software modules that support tracking, creating and managing annotations, point cloud and skeleton visualization and finally an interaction module that processes user inputs.

Users have an abstract body representation created by the skeleton provided by the "Creepy Tracker" module Sousa et al. (2017), since their real body is covered by the HMD. This system works by combining skeleton information from various Kinect cameras, allowing users to move freely in the capture space. Skeleton data is received by the application through the network, with users identifying themselves at the start-up of the application by raising their hand.

The annotation manager module is an aggregated class containing a set of modules responsible for storing and managing annotation types in the Virtual Reality environment. Each annotation class has an associated component responsible for

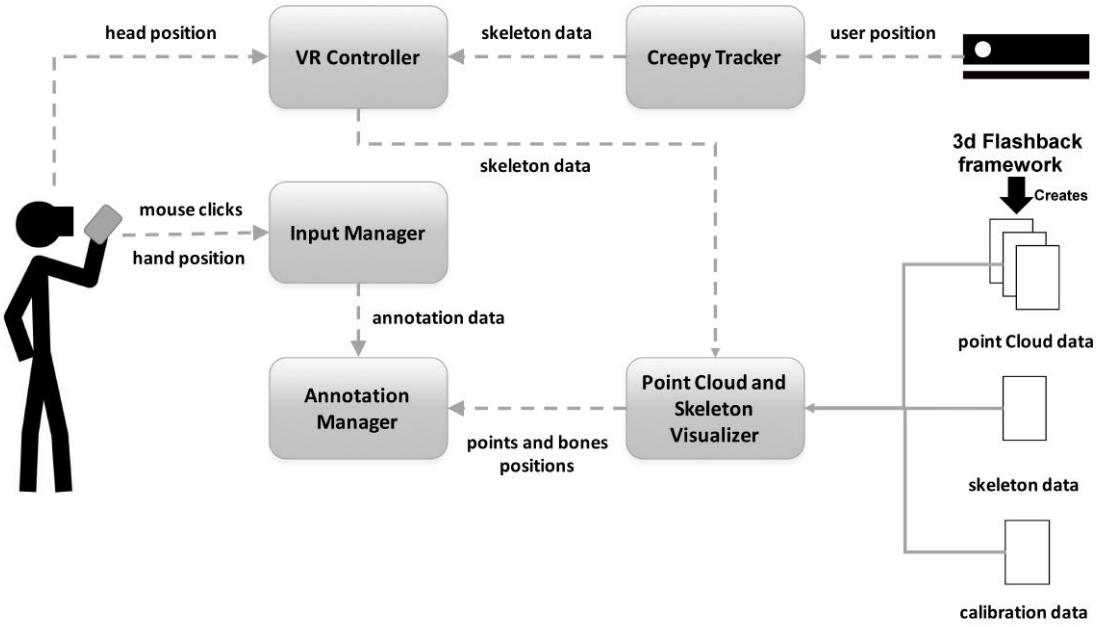


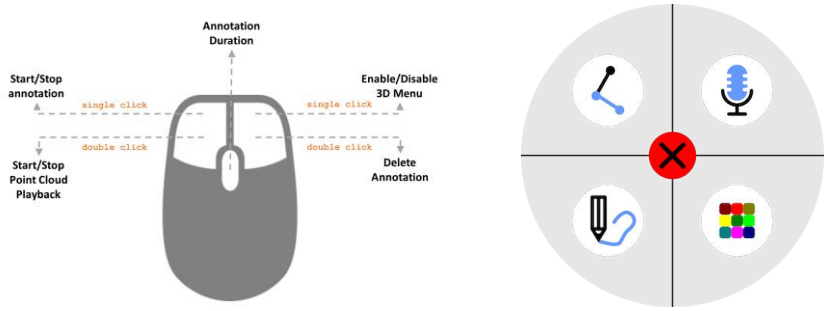
Figure 5.4: Virtual Reality Annotator modular architect diagram

processing user input and managing the 3D objects related to the annotation. In this manner extending the current software with new annotation types is straightforward and involves minimal changes to the code.

The position where the annotation should be created in the virtual world is provided by the VR Controller module, which receives and processes the skeleton data given by the Creepy Tracker and the head orientation given by the Oculus Rift. This data is also used to update the user skeleton data in the VR environment.

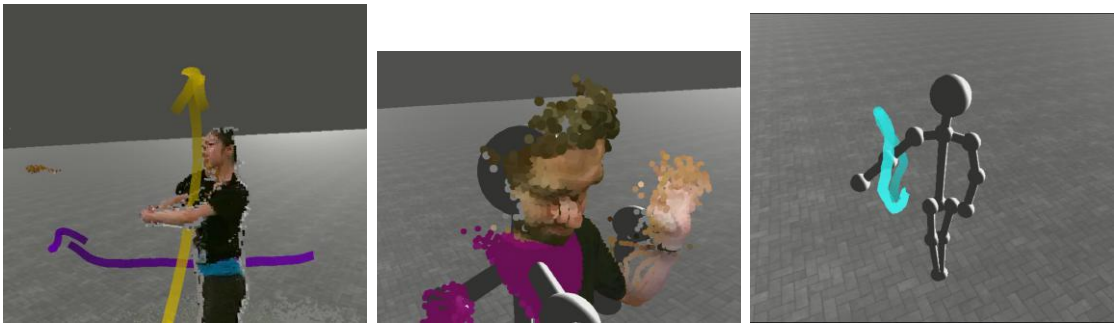
The Input Manager receives the mouse clicks (see Fig. 5.5) and hand position. Based on this data, it toggles the annotation type that is currently active. Moreover, it is responsible for visualizing and hiding the 3D menu and drawing the contextual heads-up display attached to the users' hand.

To interact with the menu, a raycast is drawn starting from the users' hand position and, when a collision is detected, the appropriate method is executed enabling or disabling the appropriate menu option. The 3D menu has five options: highlight points, speech-to-text, 3D drawing, change color, and delete (see Fig. 5.5).



(a) Mouse control schema (b) Menu

Figure 5.5: Input and interface elements



(a) 3D drawing (b) Highlight points (c) Contextualized drawing

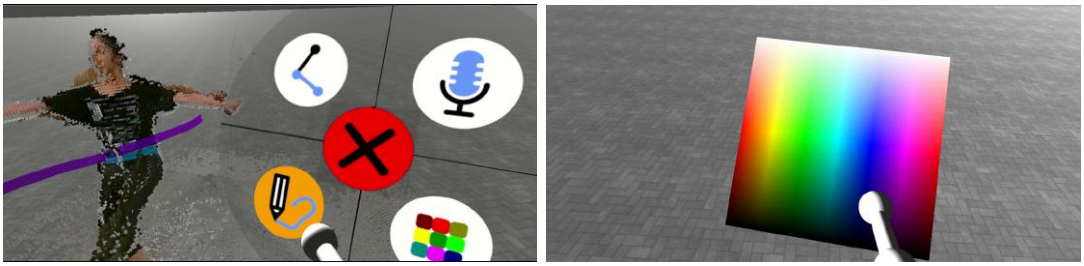
Figure 5.6: Implemented annotations/functions in our system

5.2.2.2. User interaction paradigm

Interaction with the system is performed by free navigation in the environment, and mostly based on the position of the users' dominant hand. Figure 5.5a displays the input commands of the wireless mouse . Menu interaction is performed by right clicking to open a menu in the looking direction, and selection of the current function by pointing and left clicking.

Color selection 5.7b is performed by pointing at the desired color on a RGB spectrum and left clicking. Annotations are created by holding the left mouse button and performing the desired action. For 3D drawing 5.6a the hand position is used as a virtual brush. For cloud highlighting 5.6b the same metaphor is used, except the user chooses paints the desired points instead of a general area. Finally, text annotations are created at the users' hand location.

The duration of an annotation can be adjusted by placing the dominant hand near the center of the annotation, clicking and scrolling the mouse wheel until the desired duration is reached. This is then confirmed via a second wheel click. An annotation can be deleted by a double right-click near the desired annotation.



(a) Menu Selection (b) Color picker

Figure 5.7: Examples of interaction

5.2.3. Results and Discussion

The implemented system allows users to visualize and annotate temporal point-cloud data that can be associated with skeletal information. This is particularly important in the case of highlighting 3D points of a time-sequenced point cloud.

The used datasets were casual captures in a laboratory environment, and the described dancer captures with Miyu Matsui in the Rui Lopes Graça study described in Section 1.1.

Given that there is no temporal relation between 3D points in different frames , we associate the annotations to the skeleton joints, which are updated every frame and passed to the shader that will affect the points in the vicinity of that joint. This is optional for 3D drawing and text annotations.

Some uses for this are to attach a text annotation, like a name of the person, to follow a certain subject in the video, or to create a note related to a series of actions. The same applies to drawing, which might be related to a certain body part, or simply markings on the floor. If annotations are created closer than a certain threshold to a specific skeleton joint in the frame, they are associated to that body part.

This is one of the advantages of the proposed system over previous attempts, where we are able to have both static and dynamic annotations, contextual or not, and affecting unstructured data (point cloud). Moreover, such 3D contextual annotations are only possible in the immersive VR environment, due to the nonexistent depth component in more traditional inputs.

As opposed to more complex input solutions, our system has the advantage of using known metaphors for users (painting with your hand, pointing at something you want). Also, such an embodied solution allows users to more accurately perceive and interact with space such as it is perceived in a dance studio. Allowing one

to freely navigate around the data overcomes the stated limitations of video-based teaching approaches Guest (1984).

The described system has been applied to both documenting and archiving dance choreographers' work Ribeiro et al. (2017) in the context of contemporary dance. Instead of adding annotations targeted at teaching, scholars could highlight characteristic movements or tendencies of a certain choreographer. This is a crucial application for enhancing digital cultural heritage archives, where common videos combined with text-based descriptions are not able to efficiently represent certain temporal and spatial aspects embedded in an immediate context of the documented works. An added benefit of this type of annotated data is that it can be used to develop comparative studies on different genres, styles, or periods of time in the work of choreographers and dancers.

Some of our current limitations are shared by other VR-based systems. Some users may experience motion sickness in prolonged interactions with the system, due to the fact that the markerless tracking solution applied is subject to some noise and delay as mentioned in Sousa et al. (2017). Also, the tracked space by current VR systems is limited, which is a problem if the captured area is larger than the VR tracked space, in that it restricts the users possibility to reach the position where they want to annotate. This chapter describes each one of the developed applications.

5.3. Capturing and Documenting Creative Processes in Contemporary Dance

This section is adapted from a full paper published at the International Workshop on Movement and Computing (MOCO) in 2017 [4]

A choreographer's legacy is both a historical record that resides in an archived form and a cultural memory or experience that the choreographer envisions to leave behind for future audiences, educators, and scholars (LaFrance, 2011). Historically, one of the ways in which Western theatrical dances have been preserved is through the act of score-writing. Despite the efforts of dance-notation practitioners to preserve the art of score-writing, in the latter half of the 20th century, traditional notation systems have become overshadowed by videos as a mode of documenting dances. As technological advancements have continued to alter the practice of dance

[4]Ribeiro et al. (2017)

preservation, a body of literature has emerged that illuminates alternative forms of documentation (Grammalidis et al., 2016).

Current practices for documentation and preservation are mostly focused on the development of an archive that in itself holds a structure that mirrors a choreographer's life work. In particular, it takes into account the best practices in photo and video digitization and the development of appropriate metadata standards to catalog each component of the archive material. Other relevant considerations are related to the development of a system that enhances human-computer interaction between the user and the archive. These are all significant issues that should not be neglected. Nevertheless, there still is a lack of work that targets specifically the documentation of creative processes, namely by focusing on the development of a conceptual framework where the evolution of a choreographer's imagistic universe, and how it is translated into his/her aesthetic vocabulary, can be understood.

Inspired by advances in 3D data capture and point cloud visualization techniques, this section describes how these techniques were used to capture and document the artistic work of João Fiadeiro, a Portuguese choreographer. Over the course of two decades, Fiadeiro has been developing a method he calls Composition in Real Time, Composição em Tempo Real, in Portuguese (CTR), which allows him to transmit his artistic *modus operandi* to his collaborators and to teach a series of compositional principles in international workshops and master classes (Jürgens et al., 2016). Together with Fiadeiro, we have identified a sub-set of five core concepts of his method (out of an inventory of 72 terms and expressions), which have then been used as starting-points of two improvisation sessions involving Fiadeiro's dancers and himself. Moreover, those same concepts were also the basis underpinning the development of new visualization techniques that better illustrate in an interactive system how these concepts influence the resulting final performance.

5.3.1. Documenting and Preserving Contemporary Dance

In order to better understand and inform the present by richer interpretations of the past, several actions are being taken at a global level to create affordable and efficient digital access, documentary methods analysis, and preservation services for cultural resources.

In the case of the most ephemeral and intangible art forms, such as contemporary performing arts and dance in particular, the interest in their documentation is even greater precisely because this ephemeral performative action is the ideal territory for

working within a fertile nexus of mediations. Each performance brings into play an extended series of instances: creative, technical, material, and societal. In addition, they are linked to research and education materials which also produce a vast amount of associations and may be linked to a myriad of data points. The material generated and the theoretical conclusions reached allow the exploration of new work based on existing information systems, infrastructures, and services in order to make that content available for researchers and educators in a wide range of contexts. As embodied and ephemeral practices, the performing arts comprise a fusion of cultures and practices, but their fluidity within and across borders has been characterized by a dispersed and fractured documented history. Perhaps because, in general, they are heterogeneous practices where the body is the most complex instrument to 'write down', the performing arts have produced very little in what regards "hard-copy records of themselves" (Whatley, 2013). However, this is changing enormously, both concerning the range of methods and systems currently being explored to document performing arts heritage, and in relation to the newly invigorated, creative industry initiatives within Europe.

Slightly over the last ten years, performing artists and researchers in Europe, the USA, and Australia have begun to collaborate in order to explore how digital technologies can offer new methods for documenting dance, theater, and performance in general and for increasing access to these more intangible art forms. These collaborations have resulted in novel ways to preserve and/or reconstruct existing pieces and the creative processes behind them. This is an important moment in time for performing artists in particular: by bringing together different cultures, different practices, and different disciplines, they have a significant opportunity to shape the narrative around their works with a view to creating the optimum framework for its sustainable support into the future. Moreover, recently a number of internationally known choreographers and institutions (e.g. Wayne McGregor — Random Dance, Siobhan Davies in the UK; Emio Greco—PC in the Netherlands, Rui Horta Stage Works in Portugal; the Forsythe Foundation, the Pina Bausch Foundation and the Digitaler Atlas Tanz initiative in Germany) have been proving that it is now essential for contemporary dance and performance to worry about its preservation over time. Other art forms are documented and preserved in their own medium such as painting, sculpture, and film. In the case of music and theatre, there exist universally readable documents in the form of musical notation scores and the dramaturgic script. Contemporary dance and performance, on the other hand, must develop new forms of mediation and produce new types of resources that make choreographic/compositional ideas more accessible to be studied and un-

derstood. The varieties of information-rich resources they have created (including on-line scores, dance digital archives, choreographic software agents and real-time training simulations) confirm their increasing investment in the digital preservation of contemporary creations, an interest which was previously absent (Delahunta and Shaw, 2008).

There have been until now different large-scale initiatives at the European level, such as Europeana (Eur, 2014 (accessed Jan 3, 2017), ECLAP (ECL, 2011 (accessed Jan 10, 2017) (an e-library for the performing arts), and some other specific artist-drive ones, such as the Siobhan Davies company's "RePlay Archive" (Sio, 2013 (accessed Jan 10, 2017; Whatley, 2013), where it is possible to navigate online not only through a life-long career of choreographic pieces, but through a catalog of early rehearsal footage and, where available, of filmed records of choreographies in the studio, including original rehearsal materials. The "Pina Bausch Foundation", founded after her death, is also preparing the much expected "An Invitation from Pina: The Pina Bausch Archive" (pin, 2013 (accessed Jan 10, 2017), which has developed rather ground-breaking ideas on what dance and theater documentation can be about, aside from the foundation's" more institutional archival initiatives. Also, the "K3 Zentrum für Choreographie" (k3, 2017 (Accessed: 2017-04-22) has an ongoing cooperation project called "Reflex Europe" which is focused on studying how documenting the creation process can improve learning in the context of contemporary dance.

To conclude this listing, "Motion Bank" (mot, 2010 (accessed Jan 12, 2017) was the newest Forsythe Company project (after their highly successful project "Synchronous Objects" (syn, 2013 (accessed Jan 12, 2017)), launched in 2013, and having provided a broad context for research into choreographic practice. Their main focus was on the creation of on-line digital scores in collaboration with four guest choreographers of broad international projection. However, they do not aim at an open-ended platform to accommodate inter-relations and collaborative resources between performing artists in general, as accomplished with the "Transmedia Knowledge-Base for performing arts (tbk, 2010 (accessed Jan 20, 2017), developed between 2011 and 2013, and officially launched online in June 2016.

Technologies developed for the purpose of analyzing and archiving movement in creative areas such as dance and performance are opening up new ways of exploring and reactivating specialist gestural knowledges from the past (Norman, 2016). In this manner, contextualized simulations and re-enactments of corporeal movement are providing insights into gestural skills that are otherwise being rapidly subsumed and forgotten in the wake of technological developments - notably those involving

digitization and manual or semi-automatic video annotations.

5.3.2. Composition in Real Time

Since 1995, Fiadeiro has continued to develop his Composition in Real Time (CTR) method. At first, the system was established to collaborate with fellow artists to successfully create coherent stage works. As the method was systematized over the years, not only an increasing number of artists were engaging with the CTR, but non-performers as well, such as theoreticians, academics, and researchers from such diverse fields as the social sciences (philosophy, anthropology, ethnology), economics, complex system theory, and neurosciences. The dialogue with these experts in turn helped to enrich the CTR into a full fledged compositional system, which principles can be applied in different artistic and scientific processes.

At the heart of the CTR lies a novel approach to the decision-making process. Fiadeiro observed early on in dance improvisation that performers would not dedicate as much time to evaluating their possibilities to act in a certain set of circumstances. The CTR method suggests to suspend the impulse to act immediately upon the situation one finds oneself in. Instead the performer takes the time needed to better understand the properties and possibilities inherent to the situation to make an informed decision. This decision results in an (performative) action, which is called a 'position' in the CTR. Another performer will subsequently perform the next action (or position), also using the time during suspension to make a decision which relates to the action (or position) of the performer before. In other words, a relation between the two positions (actions) is created. The third action taken by the next performer should relate to the relation established by the two actions before, so that a clear direction for the improvisation is proposed. During the cycle of vitality this direction is explored, until the improvisation comes to a natural end that all performers agree upon.

To put it simply, all participants in an emerging compositional process act collaboratively, building something by relating to each others' actions. By working in this interrelated way, it becomes evident for the participants that many possibilities exist at each moment of decision, which can be seen as "possible futures" (and pasts, for that matter). Whenever a decision has been made, the past is retroactively rewritten, that is, interpreted in a new way. Particularly this latter concept has been attracting much attention by researchers from the above-mentioned scientific domains.

5.3.3. Case Study

The main aim of our case study was to capture and visually represent a sub-set of the most representative concepts of the Real-Time Composition method. In order to do so, we captured two improvisation sessions with João Fiadeiro and seven of his dancers in two separate days. Each session had an approximate duration of three hours, and each improvisation started with the same position.

5.3.3.1. Composition in Real Time Concepts

Fiadeiro has devised a system representing his own creative process in which the underlying fundamental concepts are not static; they are organic and evolve as Fiadeiro himself transforms his own choreographic perspectives and practices based on new experiences and projects. He has also devised a version of his game to be practiced by people not accustomed to using their bodies in the same way as dancers, where objects are used on a table instead of bodies on a dance floor. In order to acquire sufficient knowledge of his method, we actively and passively participated in several workshops, including table versions and more specialized workshops targeted at expert dancers.

Following these sessions, it was important to understand how these core concepts of the CTR method related to the CTR system, as well as to the final result of an improvisation session. In particular, we wanted to understand if it were possible to visually represent his concepts using improvisation sequences from Fiadeiro's current works. With this aim, we conducted several unstructured interviews with Fiadeiro where we first focused on identifying a set of core concepts. From these interviews resulted the selection of five concepts described in Table 5.1.

The next stage focused on observing videos of Fiadeiro's past works, including rehearsals, performances, and workshops, and identifying a set of video sequences corresponding to each concept. This allowed us to have both a conceptual understanding as well as a visual representation of the concepts. We have then together with Fiadeiro developed a set of possible visualization effects for each concept. Final results are described in Section 5.3.5.

CTR Concept	Description
Position	Any performed act/ A performative act. Example: a performer lifts a chair.
Relationship	Performing a position which relates to another position. Example: another performer also lifts up a chair.
Suspension	The time a performer takes to make a decision and suspends any action.
Cycle of Vitality	The time or duration of an improvisation exploring a specific idea.
Possible Futures and Pasts	At each moment in the improvisation, a performer can choose from several possibilities of how to proceed (= futures), which will shape reinterpretations of past understandings of positions.

Table 5.1: Summary of five core Composition in Real Time concepts.

5.3.4. Motion Capture and Point Cloud Visualization

From the participation and observation of Fiadeiro's workshops, it was clear that space was a limitation, meaning we had to considered how many Microsoft Kinects would be necessary, and what was the ideal setup to guarantee data quality. To guarantee the capturing of the complete improvisation session, a wide-baseline setup was used, where each view was captured by a Kinect sensor. Each view was positioned on opposite sides of the room in order to minimize loss of data due to occlusion (see Fig. 5.8). The capture process used the framework described in Section 2.5.1.1

The remaining of the process follows the data flow presented in Figure 5.9, regarding the different parts of the 3D Flashback framework. First the color images and depth maps are converted into point cloud data. This type of data are simple Cartesian coordinates and their respective color information, therefore there is no contextual information about what is actually represented in the point cloud. At this stage, it is not yet possible to manipulate individual elements of the point clouds, such as people or objects.

The Point Cloud Library Rusu and Cousins (2011a) Euclidean Cluster Extraction algorithm was used to separate the point cloud into clusters. This algorithm basically separates unorganized point clouds into separate clusters (sets of points) based on

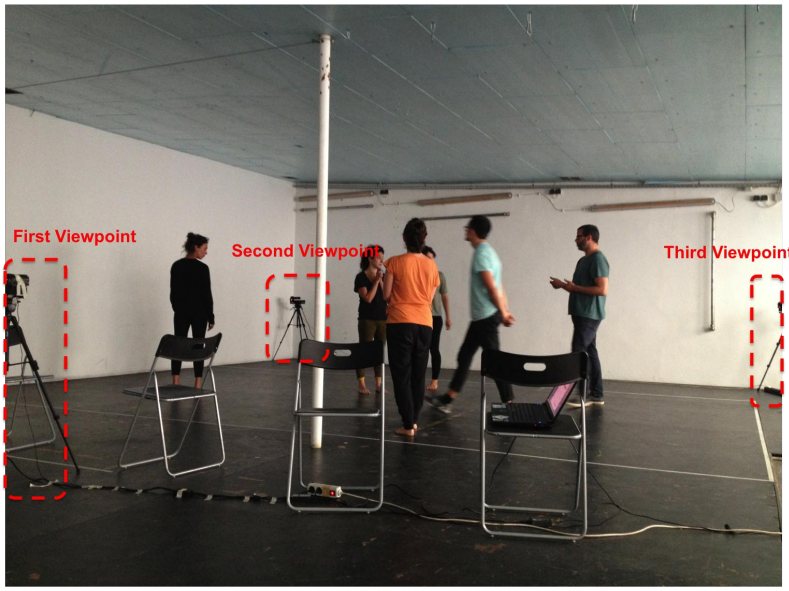


Figure 5.8: Representation of the setup used with three Kinects and three video cameras during the case-study done with João Fiadeiro and his contemporary dancers.

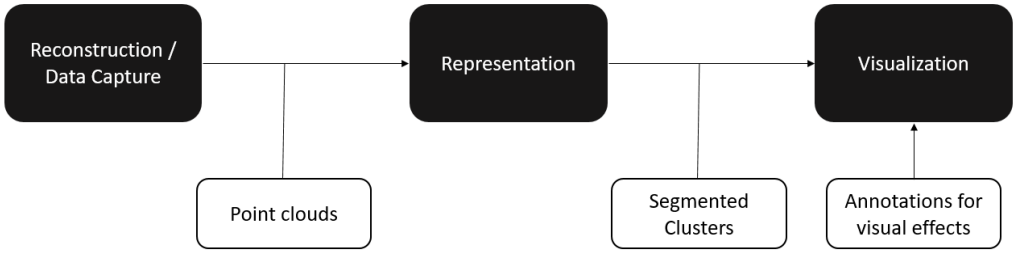


Figure 5.9: Flowchart of the application regarding to the 3D Flashback framework. It uses all the steps of the framework with some differences in the representation step, where the goal was to identify different clusters.

their distances. After the initial tests it became obvious that this information was not enough to clearly separate point clouds into clusters. Specifically, in certain situations the dancers were all concentrated in the same position (e.g. while holding the same object) and only moved slightly, tricking the algorithm to assume they became a different cluster.

To improve cluster detection we adapted the original algorithm by adding another condition to be considered besides distance. First we considered only the first frame, and for each cluster we calculated the AABB box and respective center. We then processed the remaining frames using the distance between points and also verified that the distance to the center of the AABB box of the frame before is below a certain threshold determined during the testing phase.

There are also challenges related to point cloud visualization such as loss of definition on close-ups, foreground/background confusion, and depth perception due to the absence of occlusions between the rendered points. These issues are specially noticeable on low resolution scans or background objects.

Finally, we developed a point cloud visualizer using Unity3D [5] with a simplified version of the visualization technique described in Section 4.3.2 Within this visualizer we implemented scripts which automatically apply the intended effects on to the point cloud clusters according to annotation data for the concepts of position, relation, and cycle of vitality. The remaining two concepts were implemented as different scenes in Unity3D, since they had to be manually crafted to adapt the contents of the improvisation sessions.

5.3.5. Results

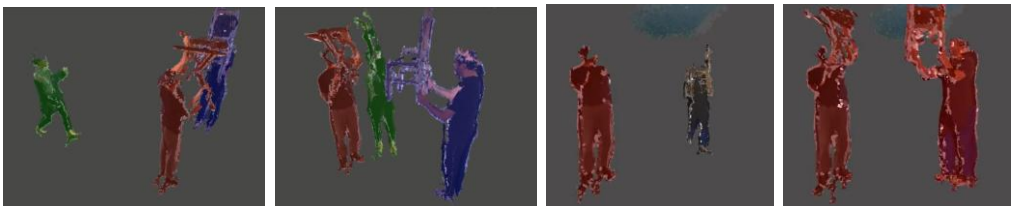
In order to illustrate each one of the concepts, short sections of data from the improvisation sessions which clearly demonstrated the selected base concepts from CTR were selected.

Since some concepts could only be well perceived by the external users through long periods of time, we combined different subsections in order to offer a shorter but sequenced visualization. To visualize the data we used a point cloud visualizer previously implemented in Unity3D. Each data set used to illustrate a particular concept was post-processed, as explained in the previous section, and then read offline by the point cloud visualizer. The software also includes a basic graphical user interface which allows the user to interact with the point cloud, specifically to navigate freely within the 3D scene, fast forward, or view each frame step by step.

The most relevant CTR core concepts, **position**, was represented through assigning different colors to elements introduced in a certain position, so that users can keep track of the development of the improvisation session. In our chosen segment, each participant entering the stage has created a new *position*. We have identified each different body through a color-based clustering algorithm, combined with video annotations Ribeiro et al. (2016), which would connect each player to a different position and an assigned color.

An example can be seen in Figure 5.10a, where the third *position* is being introduced.

[5] https://unity3d.com



(a) Two shots of the *position* example. Each participant represents a consecutive *position*

(b) Relation: The second participant slowly assimilates the color of the first as they establish a relation

Figure 5.10: Color-based representation of CTR concepts.

The participant in red entered the space holding a table in the air (first position) followed by the one in blue, who was holding a table in the air (second position), and by the one in green (third position) with a cup that is held above the table.

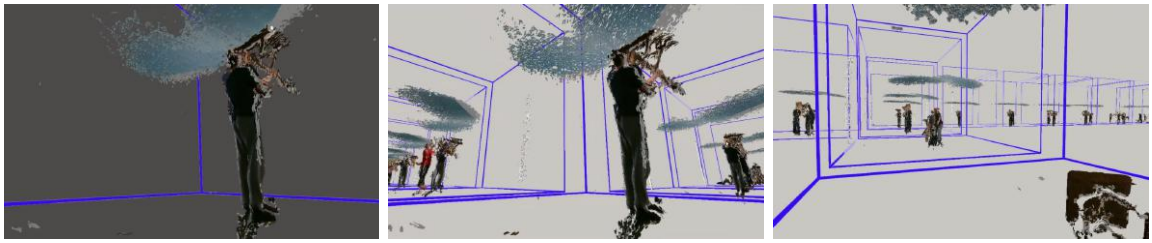
Relations formed by different elements introduced in the composition were also identified by the same color. The elements introduced in a new *position* may create a *relation* with some or all of the elements from previous positions in the scene. The establishment of newer relations was represented as a gradual assimilation of colors, as seen in Figure 5.10b. This example only contains a single *relation* between two elements: the participant holding a chair and the one holding the table, which creates a clearly identifiable relation coming from the real world. In a more complex scenario, several separate relations may co-exist at any given time. Changes in relations and positions can go by unnoticed by novices to this method, specially in longer scenarios, with more performers, or in a faster paced sequence of actions. Albeit a simple solution, color coding can clearly contribute to identifying these concepts without adding visual pollution to the actual recorded content.

The **Cycle of vitality**, i.e. the time frame during which the composition is progressing, is exemplified through the loss and gain of color. Following the CTR rationale, the end should always be postponed by allowing the creation of new relations through new *positions*. We selected a long sequence of a session performed by two participants, where we were able to see the prolongation of the *cycle of vitality* through several *positions*, as well as its approach to an end, when no new positions were being introduced (see Figure 5.11b). In this particular example, one participant decides to leave the scene as he hurts himself during the performance. The cycle was coming to an end, but another performer introduces a new position by inviting him back to the stage, using the *accident* as an element of their improvisation, therefore starting a new *cycle of vitality* from that *relation*.

The three-dimensional aspect of the captured data allows a better representation

(a) Suspension: Sequence of instances in the visualization where the concept of *possible future* appears in the background, and the camera navigates in that direction when the *suspension* ends.

(b) Cycle of vitality: The cycle ends as a performer leaves the stage. Saturation gradually comes back when another performer starts a new *position*.



(c) Possible futures and pasts: Possibilities are unveiled through the sides of the cube, where possibilities are organized in a three-dimensional space.

Figure 5.11: Sequences of the generated visualizations for CTR concepts.

of the concept of **Suspension**, which is used in the decision making process. By spatially separating present and future possible positions and manipulating the reproduction speed of a segment, we exemplify the suspension of time in which one participant is examining the current situation with her personal time being disconnected from the real time. Figure 5.11a shows our visualization of this concept. In this particular sequence, each position would add objects to the objects pile carried by the participant who had the table in the first position. At a certain moment, a critical point is reached, where it should be considered to finish the current sequence of positions, due to the fact that the first participant was not able to carry more objects. Here a *suspension* is exemplified, with the reproduction speed of the "present" (participant holding objects) slowed down, to represent the disconnect between real time and the time as perceived by the outermost participant, representing the desired decision making process in CTR. The most likely future possibility can be contemplated in the background, and when that future is "accepted" (each participant implicitly decides this is the course of action to be taken), the camera navigates

into the new present scene.

One remarkable feature of Fiadeiro's work is the concept of *rewriting the past by changing the future*, i.e. by creating new *relations*. Our last example shows **possible futures and pasts**. We recorded different improvisation sessions always repeating the first position and giving freedom to the performers from then onwards. We spatially organize the *possible futures* in a three-dimensional space. The performer is placed inside a cube, in which he has a *possible future* in each one of the sides of the cube. We navigate the camera through each one of the possible futures for new *relations*. In reality, there are countless *possible futures* for each *position*. We display all the different outcomes from starting at the same first position to exemplify this concept.

5.3.6. Discussion

Capturing and documenting contemporary dance is in no way a trivial task. This is due to the ephemeral nature of dance, which constitutes a challenge when trying to understand which dance objects are valuable resources for the research community, choreographers and dancers, and the general public. Even though it is currently common practice for choreographers or foundations/institutes to maintain some sort of archive of their works (typically comprised of photos, interviews, and videos), the same is not true when considering the documentation of creative processes. Moreover, photos and videos are static content which impose the vision of the person who photographed/recorded and edited the material. Therefore, showing but one perspective of the complex system that composes a choreographic work. These kind of static material also limits the ability to interact, experiment and explore the creative dimensions underlying a choreographic process.

Capturing dance in three dimensions supporting a viewpoint-free visualization opens up a vast range of possibilities. Primarily, dance is about movement, specifically about expression through movement. Having an interactive digital representation of movement allows us to closely observe subtle particularities that characterize a particular aesthetic language and, in this manner, preserve choreographic works within the creative context where they had been created. 3D data is a composition of graphical primitives that can be manipulated and changed to serve diverse purposes. By manipulating this kind of data it is possible to provide different viewpoints which can open up new future perspectives related to a choreographer's piece. This was the ultimate goal of our work: to provide an alternative and interactive visualization

so that future generations can create their own stories when experiencing Fiadeiros' conceptual work.

Although the proposed visualizations are specific to Fiadeiro's creative process, the methodology used to create them can be applied to other contexts. Annotating three-dimensional point cloud data is a flexible-enough basis to build different effects upon. Among the presented results, the color manipulation effects (*Position, Relation, Cycle of vitality*) may have a direct application in scenarios with different sets of concepts. Spatial organization of selected parts of a performance can also be used to demonstrate its development or evolution over longer periods of time, comparisons between similar segments, or other types of relevant concepts to the work at hand.

We should add that in this study we have only taken into account the participation of the choreographer in the design and evaluation of the system, and not the dancers'. In other words, we did not focus on the possible feedback that could be given by the dancers to improve the system, as our main focus was on the process of composition in itself, namely on the particular method of "Composition in Real Time" and not on the usability of the visualizer we have developed.

This technique has its own challenges and limitations which have to be mentioned. In particular, it is necessary to understand how a performance can be enhanced through the use of technology without compromising the dance object. Also, how this type of data can be used in a virtual environment and providing a way for the user to intuitively understand the inspiration and motivations behind that dance object.Although several affordable motion capture systems exist, their use is not at all straightforward.It is still necessary to develop specialized software to deal with this kind of data, which requires software engineers. Moreover, maintaining digitized archives requires specific maintenance and specific hardware that can often be costly.

5.4. Summary

Three different applications built upon the 3D Flashback framework were presented. Firstly, an extension to an existing 2D video annotation system has been developed, adding depth streams to multiple recording points of view of a dance performance. Each 2D video is annotated separately and, after being processed by the developed system, all annotations are combined on a single point-cloud based three-dimensional

video that allows the user to visualize the performance and its annotations from any arbitrary point of view.

We also described the Virtual Reality Annotator, which is a software tool that allows a user to visualize and annotate dynamically point cloud and skeleton data. Currently, is supports three types of annotations, namely highlight points, speech-to-text and 3D Drawing. Our tool supports dynamic annotations which is an improvement on previous existing systems.

Also, we have presented an alternative approach to archiving and visualizing the choreographic work of a performing artist, by using the work of João Fiadeiro as a case study. Not only we enable a better spatial perception of the work, but have also developed meaningful visualizations to illustrate core concepts of Fiadeiro's Composition in Real Time method.

6 Conclusions and Future Work

6.1. Conclusions

The proposed 3D Flashback framework and applications had several key aspects that needed to be solved in order to be viably used in interactive applications. All of the proposed steps were successfully implemented and described in this document. Over all the tackled problems, we focused on the single-layer data representation issue. The MVLDI showed better redundancy detection and number of generated layers in both wide-baseline and narrow baseline scenarios (Anjos et al., 2017). When compared to MV+D, we were able to save up to 49% space in a single frame on our most complex scenario, just from removing redundant data that had no impact in the visual quality of the cloud. The more input cameras are used, the higher the advantage the proposed representation brings. And the more accessible depth information is, the more relevant the MVLDI will become. These advantages were carried on to the temporal scenario, where we saw that the MVLDV saved 69% space on a simple scenario. When compared to the LDI, our approach can support wide baselines while not discarding important data, generates considerably less layers,

and is still more compact in the video scenario due to not only higher redundancy detection, but also having denser layers, which leads block matching algorithms used in video compression algorithms into better results.

Within the representation problem, we evaluated the viewpoint selection and generation problem, which creates guidelines for future research in MVLDI. One important contribution of this research was a synthetic viewpoint generation algorithm for MVLDI, which can efficiently represent any complex point cloud into images, even if the input dataset is in a point-based representation. This algorithm allows researchers to use the advantages of image-based representations in these scenarios, having access to simpler and more efficient rendering algorithms.

Stroke-based Splatting was also introduced as a novel rendering technique to visualize point clouds (dos Anjos et al., 2017). Not only it allows one to visualize this type of data with a comparable or higher visual quality than other approaches (splats, meshes), but it also applies a more efficient tangential vector estimation technique (Householder formula) which allows us to only save normal information when dealing with a video scenario, since tangents can be estimated in real time. It is also more resilient to noise, creating a less variable tangential field which can be noticed in temporal sequences.

We created a Kinect-based calibration toolkit (Sousa et al., 2017) which can be used in widely different capture scenarios, or for data streaming on interactive systems. Also, software for data capture including synchronization and segmentation were also implemented and described. Finally, compression, decompression, rendering, and integration with applications was discussed in detail, enabling future researchers to replicate and further develop the presented pipeline.

We also introduced three different applications that use the 3D Flashback framework in the context of contemporary dance teaching and digital cultural heritage. (Ribeiro et al., 2017, 2016) Using this alternative data type, we were able to go forward not only in the realism and complexity of the stored data, but also created richer interactive experiences which help choreographers and dance students to communicate about core concepts of their craft. Using virtual reality to annotate dance allows users to be in the performance space at all times, and closely interact with the visualized data. We believe this presented framework can be used to support richer interactive applications not only in dance and performing arts, but also sports, entertainment, culture, and other scenarios where spatial perception and three-dimensional data is important.

6.2. Future Work

Regarding data representation, our research on viewpoint evaluation order still needs to be improved. The impact of this step increases with the number of input cameras used. Our results showed that search-based approaches are overall more successful, however more research has to be done regarding heuristics for viewpoint selection, and a quasi-complete search algorithm needs to be implemented, opposed to both the greedy approach presented in this paper, and the complete search (*optimal*) which has a very high complexity for video scenarios. Our results also confirmed that selecting a viewpoint directed according to the normal of the biggest surfaces in the point cloud can be a successful approach, depending highly on the results of the clustering algorithm applied. Specialized techniques for detecting surfaces on highly fragmented point clouds need to be developed, or novel approaches for residual data need to be applied.

Also, the impact of different viewpoint selection approaches in video compression has to be evaluated. Two main questions can be raised related to this topic.

- What are the advantages of constantly re-evaluating a better viewpoint for each layer over only performing this process at certain points of the stream?
- Should different viewpoints be encoded in the same temporal compressed layer, or keeping separate layers for each viewpoint is the best approach?

Correctly answering these questions will further solidify the MVLDI as a compact and efficient representation for multiview videos. Finally, approaches for automatic estimation of the redundancy threshold can be researched for scenarios where the desired sampling of the data needs to adapt to the content.

Several improvements to the Stroke-Based Splatting technique can be introduced in the context of rendering video data. Namely, estimating normal vectors from the depth images in real time can decrease even further the amount of data necessary for rendering. This can be done by calculating the derivatives from neighboring pixels in the depth component of individual layers of the MVLDI. Moreover, an alternative color blending algorithm has to be developed in order to be applied in a video context, where A-buffering is too heavy.

One future research line will be exploring hole filling algorithms for surface completion using the MVLDV representation. We believe we are able to more easily identify plain surfaces on the scene on image-space, treating them on a simplified

two-dimensional space. By encoding surfaces with similar normal values into the same layers, we can more easily detect holes and find boundaries to work upon.

Integrating the 3D Flashback framework in different output devices such as augmented reality glasses will also help us create novel applications, and solve different real world problems. Overall, we believe our contributions to the VBR field will allow researchers to develop novel wide-baseline applications and experiences in a world where depth information is more and more accessible.

Bibliography

Motion Bank, 2010 (accessed Jan 12, 2017). http://motionbank.org/en.

Transmedia Knowledge-Base for performing arts, 2010 (accessed Jan 20, 2017). http://tkb.fcsh.unl.pt/content/knowledge-base-performing-arts.

The ECLAP e-library of performing arts, 2011 (accessed Jan 10, 2017). http://labs.europeana.eu/apps/eclap.

The Choreographic Archive of Siobhan Davies Dance, 2013 (accessed Jan 10, 2017). http://www.siobhandaviesreplay.com.

The Pina Bausch Archive, 2013 (accessed Jan 10, 2017). http://www.pinabausch.org/en/archive.

Synchronous Objects, 2013 (accessed Jan 12, 2017). http://synchronousobjects.osu.edu.

Europeana Project, 2014 (accessed Jan 3, 2017). http://www.europeana.eu/portal/en.

K3 – Centre for Choreography - REFLEX Europe, 2017 (Accessed: 2017-04-22). http://k3-hamburg.de/en/koopprojekte.

A. Abramov, K. Pauwels, J. Papon, F. Worgotter, and B. Dellen. Depth-supported real-time video segmentation with the kinect. In *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*, pages 457–464, Jan 2012. doi: 10.1109/WACV.2012.6163000.

A. C. Addison and M. Gaiani. Virtualized architectural heritage: new tools and techniques. *IEEE MultiMedia*, 7(2):26–31, Apr 2000. ISSN 1070-986X. doi: 10.1109/93.848422.

Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112,

October 2011. ISSN 0001-0782. doi: 10.1145/2001269.2001293. URL http://doi.acm.org/10.1145/2001269.2001293.

Naveed Ahmed and Imran Nazir Junejo. A system for 3d video acquisition and spatio-temporally coherent 3d animation reconstruction using multiple rgb-d cameras. *Proceedings of IJSIP*, 6(2):113–128, 2013.

M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *Visualization and Computer Graphics, IEEE Transactions on*, 9(1):3–15, Jan 2003. ISSN 1077-2626. doi: 10.1109/TVCG.2003.1175093.

S. Alpert, M. Galun, A. Brandt, and R. Basri. Image segmentation by probabilistic bottom-up aggregation and cue integration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(2):315–327, Feb 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.130.

Rafael dos Anjos, João Madeiras Pereira, José António Gaspar, and Carla Fernandes. Multiview layered depth image. *Journal of WSCG*, 25(2):115–122, 2017.

P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33 (5):898–916, May 2011. ISSN 0162-8828. doi: 10.1109/TPAMI.2010.161.

Yoel Arieli, Barak Freedman, Meir Machline, and Alexander Shpunt. Depth mapping using projected patterns, April 3 2012. US Patent 8,150,142.

Naoyuki Awano, Koji Nishio, and Ken-ichi Kobori. Interactive stipple rendering for point clouds. *WSCG 2010: Communication Papers Proceedings: 18th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision in co-operation with EUROGRAPHICS*, pages 193–198, 2010.

Felipe Bacim, Mahdi Nabiyouni, and Doug A Bowman. Slice-n-swipe: A free-hand gesture user interface for 3d point cloud annotation. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 185–186. IEEE, 2014.

Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, MichaelJ. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. ISSN 0920-5691. doi: 10.1007/s11263-010-0390-2. URL http://dx.doi.org/10.1007/s11263-010-0390-2.

Luca Ballan, Gabriel J. Brostow, Jens Puwein, and Marc Pollefeys. Unstructured video-based rendering: interactive exploration of casually captured videos. *ACM Trans. Graph.*, 29:87:1–87:11, July 2010. ISSN 0730-0301. doi: http://doi.acm.org/10.1145/1778765.1778824. URL http://doi.acm.org/10.1145/1778765.1778824.

J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques.

International Journal of Computer Vision, 12(1):43–77, 1994. ISSN 0920-5691. doi: 10.1007/BF01420984. URL http://dx.doi.org/10.1007/BF01420984.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2007.09.014. URL http://www.sciencedirect.com/science/article/pii/S1077314207001555. Similarity Matching in Computer Vision and Multimedia.

Armand Belmares-Sarabia and Stanley J Chayka. Video color detector and chroma key device and method, March 7 1989. US Patent 4,811,084.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4): 509–522, Apr 2002. ISSN 0162-8828. doi: 10.1109/34.993558.

O Ben Ahmad, M Mejdoub, and C Ben Amar. Sift accordion: A space-time descriptor applied to human action recognition. *World Academy of Science, Engineering and Technology*, 2011.

M. Bertero, T.A. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76(8):869–889, 1988. ISSN 0018-9219. doi: 10.1109/5.5962.

PaulJ. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1 (2):127–152, 1988. ISSN 0932-8092. doi: 10.1007/BF01212277. URL http://dx.doi.org/10.1007/BF01212277.

R. Borgo, M. Chen, B. 1Daubney, E. Grundy, G. Heidemann, B. Höferlin, M. Höferlin, H. Leitte, D. Weiskopf, and X. Xie. State of the art report on video-based graphics and video visualization. *Computer Graphics Forum*, 31(8):2450–2477, 2012. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2012.03158.x. URL http://dx.doi.org/10.1111/j.1467-8659.2012.03158.x.

Mario Botsch, Andreas Wiratanaya, and Leif Kobbelt. Efficient high quality rendering of point sampled geometry. In *Proceedings of the 13th Eurographics Workshop on Rendering*, EGRW '02, pages 53–64, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association. ISBN 1-58113-534-3. URL http://dl.acm.org/citation.cfm?id=581896.581904.

Mario Botsch, Alexander Hornung, Matthias Zwicker, and Leif Kobbelt. High-Quality Surface Splatting on Today's GPUs. In Marc Alexa, Szymon Rusinkiewicz, Mark Pauly, and Matthias Zwicker, editors, *Eurographics Symposium on Point-Based Graphics (2005)*. The Eurographics Association, 2005. ISBN 3-905673-20-7. doi: 10.2312/SPBG/SPBG05/017-024.

Thierry Bouwmans, Fida El Baf, Bertrand Vachon, et al. Statistical background modeling

for foreground detection: A survey. *Handbook of Pattern Recognition and Computer Vision*, pages 181–199, 2010.

E. Boyer, A. M. Bronstein, M. M. Bronstein, B. Bustos, T. Darom, R. Horaud, I. Hotz, Y. Keller, J. Keustermans, A. Kovnatsky, R. Litman, J. Reininghaus, I. Sipiran, D. Smeets, P. Suetens, D. Vandermeulen, A. Zaharescu, and V. Zobel. Shrec 2011: Robust feature detection and description benchmark. In *Proceedings of the 4th Eurographics Conference on 3D Object Retrieval*, EG 3DOR'11, pages 71–78, Aire-la-Ville, Switzerland, Switzerland, 2011. Eurographics Association. ISBN 978-3-905674-31-6. doi: 10.2312/3DOR/3DOR11/071-078. URL `http://dx.doi.org/10.2312/3DOR/3DOR11/071-078`.

Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005. ISSN 0920-5691. doi: 10.1023/B:VISI.0000045324.43199.43.

S. Brutzer, B. Hoferlin, and G. Heidemann. Evaluation of background subtraction techniques for video surveillance. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1937–1944, 2011. doi: 10.1109/CVPR.2011.5995508.

L. Bruzzone and D.F. Prieto. Automatic analysis of the difference image for unsupervised change detection. *Geoscience and Remote Sensing, IEEE Transactions on*, 38(3):1171–1182, 2000. ISSN 0196-2892. doi: 10.1109/36.843009.

Diogo Cabral, João G. Valente, Urândia Aragão, Carla Fernandes, and Nuno Correia. Evaluation of a multimodal video annotator for contemporary dance. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 572–579, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1287-5. doi: 10.1145/2254556.2254663. URL `http://doi.acm.org/10.1145/2254556.2254663`.

Ricardo Cabral, Fernando De la Torre, Joao Paulo Costeira, and Alexandre Bernardino. Matrix completion for weakly-supervised multi-label image classification. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):121–135, 2015.

M. Camplani, T. Mantecon, and L. Salgado. Depth-color fusion strategy for 3-d scene modeling with kinect. *Cybernetics, IEEE Transactions on*, 43(6):1560–1571, Dec 2013. ISSN 2168-2267. doi: 10.1109/TCYB.2013.2271112.

Massimo Camplani and Luis Salgado. Background foreground segmentation with rgb-d kinect data: An efficient combination of classifiers. *Journal of Visual Communication and Image Representation*, 25(1):122 – 136, 2014. ISSN 1047-3203. doi: http://dx.doi.org/10.1016/j.jvcir.2013.03.009. URL `http://www.sciencedirect.com/science/article/pii/S1047320313000497`. Visual Understanding and Applications with RGB-D Cameras.

Antonio Camurri, Katerina El Raheb, Oshri Even-Zohar, Yannis Ioannidis, Amalia Markatzi, Jean-Marc Matos, Edwin Morley-Fletcher, Pablo Palacio, Muriel Romero, Augusto Sarti, Stefano Di Pietro, Vladimir Viro, and Sarah Whatley. Wholodance: Towards a methodology for selecting motion capture data across different dance learning practice. In *Proceedings of the 3rd International Symposium on Movement and Computing*, MOCO '16, pages 43:1–43:2, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4307-7. doi: 10.1145/2948910.2948912. URL `http://doi.acm.org/10.1145/2948910.2948912`.

J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 67–76, New York, NY, USA, 2001. ACM. ISBN 1-58113-374-X. doi: 10.1145/383259.383266.

Joel Carranza, Christian Theobalt, Marcus A. Magnor, and Hans-Peter Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 22(3):569–577, July 2003. ISSN 0730-0301. doi: 10.1145/882262.882309. URL `http://doi.acm.org/10.1145/882262.882309`.

Jacky CP Chan, Howard Leung, Jeff KT Tang, and Taku Komura. A virtual reality dance training system using motion capture technology. *IEEE Transactions on Learning Technologies*, 4(2):187–195, 2011.

Jing Chen and Axel Pinz. Structure and motion by fusion of inertial and vision-based tracking. *Digital Imaging in Media and Education (W. Burger and J. Scharinger, eds.)*, 179:55–62, 2004.

Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 279–288, New York, NY, USA, 1993. ACM. ISBN 0-89791-601-8. doi: 10.1145/166117.166153. URL `http://doi.acm.org/10.1145/166117.166153`.

Yizong Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799, 1995.

ChoreoPro. Dance designer, 2014. URL `https://www.choreopro.com`.

Credo Interactive. Danceforms. web. `http://charactermotion.com/products/danceforms/`, 2014.

Marco Crocco, Cosimo Rubino, and Alessio Del Bue. Structure from motion with objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Erik Cuevas, Daniel Zaldivar, Marco Pérez-Cisneros, Humberto Sossa, and Valentín Osuna. Block matching algorithm for motion estimation based on artificial bee colony (abc). *Applied Soft Computing*, 13(6):3047 – 3059, 2013. ISSN 1568-4946. doi: http://dx.doi.org/10.1016/j.asoc.2012.09.020. Swarm intelligence in image and video processing.

Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237269. URL `http://doi.acm.org/10.1145/237170.237269`.

Cassidy J. Curtis, Sean E. Anderson, Joshua E. Seims, Kurt W. Fleischer, and David H. Salesin. Computer-generated watercolor. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 421–430, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258896.

I. Daribo and H. Saito. A novel inpainting-based layered depth video for 3dtv. *Broadcasting, IEEE Transactions on*, 57(2):533–541, June 2011. ISSN 0018-9316. doi: 10.1109/TBC.2011.2125110.

Alessio Del Bue and Lourdes Agapito. Non-rigid stereo factorization. *International Journal of Computer Vision*, 66(2):193–207, 2006. ISSN 0920-5691. doi: 10.1007/s11263-005-3958-5. URL `http://dx.doi.org/10.1007/s11263-005-3958-5`.

Scott Delahunta and Norah Zuniga Shaw. Choreographic resources agents, archives, scores and installations. *Performance Research*, 13(1):131–133, 2008.

L. Di Stefano, S. Mattoccia, and M. Mola. A change-detection algorithm based on structure and colour. In *Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on*, pages 252–259, July 2003. doi: 10.1109/AVSS.2003.1217929.

L. Do, G. Bravo, S. Zinger, and P.H.N. de With. Gpu-accelerated real-time free-viewpoint dibr for 3dtv. *Consumer Electronics, IEEE Transactions on*, 58(2):633–640, May 2012. ISSN 0098-3063. doi: 10.1109/TCE.2012.6227470.

Rafael Kuffner dos Anjos, Claudia Sofia Ribeiro, Daniel Simões Lopes, and João Madeiras Pereira. Stroke-based splatting: an efficient multi-resolution point cloud visualization technique. *The Visual Computer*, Jul 2017. ISSN 1432-2315. doi: 10.1007/s00371-017-1420-7. URL `https://doi.org/10.1007/s00371-017-1420-7`.

Yong Duan, Mingtao Pei, and Yucheng Wang. Probabilistic depth map fusion of kinect and stereo in real-time. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, pages 2317–2322, Dec 2012. doi: 10.1109/ROBIO.2012.6491315.

David Eberly. Computing orthonormal sets in 2d, 3d, and 4d. Geometric Tools. LLC, 2016.

Carlos HernÃ¡ndez Esteban and Francis Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367 – 392, 2004. ISSN 1077-3142. doi: 10.1016/j.cviu.2004.03.016. ¡ce:title¿Special issue on model-based and image-based 3D scene representation for interactive visualization¡/ce:title¿.

Remondino Fabio et al. From point cloud to surface: the modeling and visualization problem. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(5):W10–w10, 2003.

Shufei Fan and Frank P. Ferrie. Photo hull regularized stereo. *Image and Vision Computing*, 28(4):724 – 730, 2010. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2008.10.008.

João Fiadeiro. *Wissen in Bewegung. Perspektiven der künstlerischen und wissenschaftlichen Forschung im Tanz*, chapter Wenn Du das nicht weißt, warum fragst Du dann?, pages 103–112. transcript Verlag, Bielefeld, 2007.

Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1362 –1376, aug. 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.161.

J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H. P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1746–1753, June 2009. doi: 10.1109/CVPR.2009.5206755.

Marcel Germann, Alexander Hornung, Richard Keiser, Remo Ziegler, Stephan Würmlin, and Markus Gross. Articulated billboards for video-based rendering. *Computer Graphics Forum*, 29(2):585–594, 2010. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2009.01628.x. URL `http://dx.doi.org/10.1111/j.1467-8659.2009.01628.x`.

M Ghanbari. The cross-search algorithm for motion estimation [image coding]. *Communications, IEEE Transactions on*, 38(7):950–953, 1990. ISSN 0090-6778. doi: 10.1109/26.57512.

M. Goesele, B. Curless, and S.M. Seitz. Multi-view stereo revisited. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2402–2409, 2006. doi: 10.1109/CVPR.2006.199.

Bruce Gooch, Greg Coombe, and Peter Shirley. Artistic vision: painterly rendering using computer vision techniques. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 83–ff. ACM, 2002. doi: 10.1145/508543.508545.

Google. Project tango. https://www.google.com/atap/projecttango/project, 10 2014.

Patrik Goorts, Cosmin Ancuti, Maarten Dumont, Sammy Rogmans, and Philippe Bekaert. Real-time video-based view interpolation of soccer events using depth-selective plane sweeping. In *Proceedings of the Eight International Conference on Computer Vision Theory and Applications*, 2013. ISBN 978-989-8565-48-8.

M Gopi, Shankar Krishnan, and Cláudio T Silva. Surface reconstruction based on lower dimensional localized delaunay triangulation. In *Computer Graphics Forum*, volume 19, pages 467–478, 2000.

Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 43–54, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237200. URL `http://doi.acm.org/10.1145/237170.237200`.

Amanda Starling Gould. Invisible visualities: Augmented reality art and the contemporary media ecology. *Convergence*, 20(1):25–32, 2014.

N Grammalidis, K Dimitropoulos, F Tsalakanidou, A Kitsikidis, P Roussel, B Denby, P Chawah, L Buchman, S Dupont, S Laraba, et al. The i-treasures intangible cultural heritage dataset. In *Proceedings of the 3rd International Symposium on Movement and Computing*, page 23. ACM, 2016.

Ann Hutchinson Guest. *Dance notation: The process of recording movement on paper.* New York: Dance Horizons, 1984.

Paul Haeberli. Paint by numbers: Abstract image representations. *SIGGRAPH Comput. Graph.*, 24(4):207–214, September 1990. ISSN 0097-8930. doi: 10.1145/97880.97902.

Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, volume 15, page 50. Manchester, UK, 1988.

N. Hasler, B. Rosenhahn, T. Thormahlen, M. Wand, J. Gall, and H.-P. Seidel. Markerless motion capture with unsynchronized moving cameras. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 224–231, June 2009. doi: 10.1109/CVPR.2009.5206859.

Stefan Hauswiesner, Matthias Straka, and Gerhard Reitmayr. Coherent image-based rendering of real-world objects. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 183–190, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0565-5. doi: 10.1145/1944745.1944776. URL `http://doi.acm.org/10.1145/1944745.1944776`.

Christopher G. Healey, Laura Tateosian, James T. Enns, and Mark Remple. Perceptually based brush strokes for nonphotorealistic visualization. *ACM Trans. Graph.*, 23(1): 64–96, January 2004. ISSN 0730-0301. doi: 10.1145/966131.966135.

J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, March 2015. ISSN 0162-8828. doi: 10.1109/TPAMI.2014.2345390.

Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 453–460, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. doi: 10.1145/280814.280951.

Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *ARTIFICAL INTELLIGENCE*, 17:185–203, 1981.

Hai Huang, Claus Brenner, and Monika Sester. A generative statistical approach to automatic 3d building roof reconstruction from laser scanning data. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 79(0):29 – 43, 2013. ISSN 0924-2716. doi: http://dx.doi.org/10.1016/j.isprsjprs.2013.02.004.

Yu-Wen Huang, Ching-Yeh Chen, Chen-Han Tsai, Chun-Fu Shen, and Liang-Gee Chen. Survey on block matching motion estimation algorithms and architectures with new results. *Journal of VLSI signal processing systems for signal, image and video technology*, 42(3):297–320, 2006. ISSN 0922-5773. doi: 10.1007/s11265-006-4190-4. URL `http://dx.doi.org/10.1007/s11265-006-4190-4`.

James Imber, Marco Volino, Jean-Yves Guillemaut, Simon Fenney, and Adrian Hilton. Free-viewpoint video rendering for mobile devices. In *Proceedings of the 6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications*, MIRAGE '13, pages 11:1–11:8, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2023-8. doi: 10.1145/2466715.2466726. URL `http://doi.acm.org/10.1145/2466715.2466726`.

Itseez. Open source computer vision library. `https://github.com/itseez/opencv`, 2015.

Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047270. URL `http://doi.acm.org/10.1145/2047196.2047270`.

Stuart James, Manuel J. Fonseca, and John Collomosse. Reenact: Sketch based choreographic design from archival dance footage. In *Proceedings of International Conference on Multimedia Retrieval*, ICMR '14, pages 313:313–313:320, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2782-4. doi: 10.1145/2578726.2578766. URL `http://doi.acm.org/10.1145/2578726.2578766`.

A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1943–1950, June 2010. doi: 10.1109/CVPR.2010.5539868.

Stephan Jürgens, Francisco Henriques, and Carla Fernandes. Re-constructing the choreographic studio of joão fiadeiro through animated infographic films. *PARtake: The Journal of Performance as Research*, 1(1):3, 2016.

T. Kanade. Carnegie mellon goes to the super bowl. http://www.ri.cmu.edu/events/sb35/tksuperbowl.html., 2001.

Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. *ACM Trans. Graph.*, 26(3), July 2007. ISSN 0730-0301. doi: 10.1145/1276377.1276407. URL `http://doi.acm.org/10.1145/1276377.1276407`.

Hiroaki Kawata, Alexandre Gouaillard, and Takashi Kanai. Interactive point-based painterly rendering. In *Proceedings of the 2004 International Conference on Cyberworlds*, CW '04, pages 293–299, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2140-1. doi: 10.1109/CW.2004.42.

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. ISBN 3-905673-36-3.

J. Kilner, J. Starck, A. Hilton, and O. Graii. Dual-mode deformable models for free-viewpoint video of sports events. In *3-D Digital Imaging and Modeling, 2007. 3DIM '07. Sixth International Conference on*, pages 177 –184, aug. 2007. doi: 10.1109/3DIM.2007.22.

W. S. Kim, A. Ortega, P. Lai, and D. Tian. Depth map coding optimization using rendered view distortion for 3d video coding. *IEEE Transactions on Image Processing*, 24(11): 3534–3545, Nov 2015. ISSN 1057-7149. doi: 10.1109/TIP.2015.2447737.

Michael Kipp. Anvil: The video annotation research tool. *Handbook of Corpus Phonology. Oxford University Press, Oxford (to appear, 2011)*, 2010.

S. Kirshanthan, L. Lajanugen, P.N.D. Panagoda, L.P. Wijesinghe, D.V.S.X. De Silva, and A.A. Pasqual. Layered depth image based hevc multi-view codec. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Ryan McMahan, Jason Jerald, Hui Zhang, StevenM. Drucker, Chandra Kambhamettu, Maha El Choubassi, Zhigang Deng, and Mark Carlson, editors, *Advances in Visual Computing*, volume 8888 of *Lecture Notes in Computer Science*, pages 376–385. Springer International Publishing, 2014. ISBN 978-3-319-14363-7. doi: 10.1007/978-3-319-14364-4_36. URL `http://dx.doi.org/10.1007/978-3-319-14364-4_36`.

ITARU Kitahara, HIDEO Saito, Shinji Akimichi, Tooru Ono, Yuichi Ohta, and Takeo Kanade. Large-scale virtualized reality. *Computer Vision and Pattern Recognition, Technical Sketches*, 2001.

K. Klasing, D. Althoff, D. Wollherr, and M. Buss. Comparison of surface normal estimation methods for range sensing applications. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3206–3211, May 2009. doi: 10.1109/ROBOT.2009.5152493.

Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Comput. Graph.*, 28(6):801–814, December 2004. ISSN 0097-8493. doi: 10.1016/j.cag.2004.08.009. URL `http://dx.doi.org/10.1016/j.cag.2004.08.009`.

T Koga. Motion-compensated interframe coding for video conferencing. In *Proc. NTC'81*, pages 5–3, 1981.

Ravikrishna Kolluri, Jonathan Richard Shewchuk, and James F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pages 11–21, New York, NY, USA, 2004. ACM. ISBN 3-905673-13-4. doi: 10.1145/1057432.1057434.

Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In Anders Heyden, Gunnar Sparr, Mads Nielsen, and Peter Johansen, editors, *Computer Vision — ECCV 2002*, volume 2352 of *Lecture Notes in Computer Science*, pages 82–96. Springer Berlin Heidelberg, 2002. ISBN 978-3-540-43746-8. doi: 10.1007/3-540-47977-5_6. URL `http://dx.doi.org/10.1007/3-540-47977-5_6`.

Johannes Kopf, Fabian Langguth, Daniel Scharstein, Richard Szeliski, and Michael Goesele. Image-based rendering in the gradient domain. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013)*, 32(6):to appear, 2013.

Anestis Koutsoudis, Blaž Vidmar, George Ioannakis, Fotis Arnaoutoglou, George Pavlidis, and Christodoulos Chamzas. Multi-image 3d reconstruction data evaluation. *Journal of Cultural Heritage*, 15(1):73 – 79, 2014. ISSN 1296-2074. doi: http://dx.doi.org/10.1016/j.culher.2012.12.003.

Claudia Kuster, Jean-Charles Bazin, Cengiz Öztireli, Teng Deng, Tobias Martin, Tiberiu Popa, and Markus Gross. Spatio-temporal geometry fusion for multiple hybrid cameras using moving least squares surfaces. *Computer Graphics Forum*, 33(2):1–10, 2014. ISSN 1467-8659. doi: 10.1111/cgf.12285. URL `http://dx.doi.org/10.1111/cgf.12285`.

Matthew Kyan, Guoyu Sun, Haiyan Li, Ling Zhong, Paisarn Muneesawang, Nan Dong, Bruce Elder, and Ling Guan. An approach to ballet dance training through ms kinect and visualization in a cave virtual reality environment. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(2):23, 2015.

F. Lafarge, R. Keriven, M. Bredif, and Hoang-Hiep Vu. A hybrid multiview stereo algorithm for modeling urban scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):5–17, Jan 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.84.

Cheryl LaFrance. Choreographers' archives: Three case studies in legacy preservation. *Dance Chronicle*, 34(1):48–76, 2011.

S Lee and Y Ho. Real-time stereo view generation using kinect depth camera. *APSIPA ASC*, pages 1–4, 2011.

J. Lei, S. Li, C. Zhu, M. T. Sun, and C. Hou. Depth coding based on depth-texture motion and structure similarities. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(2):275–286, Feb 2015. ISSN 1051-8215. doi: 10.1109/TCSVT.2014.2335471.

Guannan Li, Chenglei Wu, Carsten Stoll, Yebin Liu, Kiran Varanasi, Qionghai Dai, and Christian Theobalt. Capturing relightable human performances under general uncontrolled illumination. *Computer Graphics Forum*, 32(2pt3):275–284, 2013. ISSN 1467-8659. doi: 10.1111/cgf.12047. URL `http://dx.doi.org/10.1111/cgf.12047`.

Viz Libero. Viz libero: Sports broadcasting redefined. http://www.vizrt.com/products, 2014.

Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998. ISSN 0920-5691. doi: 10.1023/A:1008045108935.

Peter Litwinowicz. Processing images and video for an impressionist effect. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 407–414, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258893.

Yebin Liu, Qionghai Dai, and Wenli Xu. A point-cloud-based multiview stereo algorithm for free-viewpoint video. *Visualization and Computer Graphics, IEEE Transactions on*, 16(3):407–418, 2010. ISSN 1077-2626. doi: 10.1109/TVCG.2009.88.

D. S. Lopes, M. T. Silva, and J. A. AmbróSio. Tangent vectors to a 3-d surface normal: a geometric tool to find orthogonal vectors based on the householder transformation. *Comput. Aided Des.*, 45(3):683–694, March 2013. ISSN 0010-4485. doi: 10.1016/j.cad.2012.11.003.

Daniel S. Lopes, Miguel T. Silva, Jorge A. Ambrósio, and Paulo Flores. A mathematical framework for rigid contact detection between quadric and superquadric surfaces. *Multibody System Dynamics*, 24(3):255–280, 2010. ISSN 1573-272X. doi: 10.1007/s11044-010-9220-0.

DavidG. Lowe. Distinctive image features from scale-invariant keypoints. *International*

Journal of Computer Vision, 60(2):91–110, 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94.

P. Lubos, R. Beimler, M. Lammers, and F. Steinicke. Touching the cloud: Bimanual annotation of immersive point clouds. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 191–192, March 2014. doi: 10.1109/3DUI.2014.6798885.

Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

C. Maes, T. Fabry, J. Keustermans, D. Smeets, P. Suetens, and D. Vandermeulen. Feature detection on 3d face surfaces for pose normalisation and recognition. In *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference on*, pages 1–6, Sept 2010. doi: 10.1109/BTAS.2010.5634543.

M.A. Magnor. *Video-based rendering*. Ak Peters Series. A K Peters, 2005. ISBN 9781568812441. URL `http://books.google.com.br/books?id=RbWzOCocpbMC`.

Aditi Majumder and Meenakshisundaram Gopi. Hardware accelerated real time charcoal rendering. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 59–66. ACM, 2002. doi: 10.1145/508539.508541.

Wojciech Matusik, Chris Buehler, Ramesh Raskar, Steven J. Gortler, and Leonard McMillan. Image-based visual hulls. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 369–374, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344951. URL `http://dx.doi.org/10.1145/344779.344951`.

Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 39–46, New York, NY, USA, 1995. ACM. ISBN 0-89791-701-4. doi: 10.1145/218380.218398. URL `http://doi.acm.org/10.1145/218380.218398`.

Barbara J. Meier. Painterly rendering for animation. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 477–484, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237288.

P. Merkle, A. Smolic, K. Muller, and T. Wiegand. Multi-view video plus depth representation and coding. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 1, pages I – 201–I – 204, Sept 2007. doi: 10.1109/ICIP.2007.4378926.

P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Müller, P.H.N. de With, and T. Wiegand. The effects of multiview depth video compression on multiview rendering. *Signal Processing: Image Communication*, 24(1-2):73 – 88, 2009. ISSN 0923-5965. doi:

http://dx.doi.org/10.1016/j.image.2008.10.010. URL http://www.sciencedirect.com/science/article/pii/S0923596508001161. Special issue on advances in three-dimensional television and video.

P. Merkle, K. Müller, D. Marpe, and T. Wiegand. Depth intra coding for 3d video based on geometric primitives. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(3):570–582, March 2016. ISSN 1051-8215. doi: 10.1109/TCSVT.2015.2407791.

D.L. Mills. Internet time synchronization: the network time protocol. *Communications, IEEE Transactions on*, 39(10):1482–1493, Oct 1991. ISSN 0090-6778. doi: 10.1109/26.103043.

E.R. Moghaddam, J. Sadeghi, and F.F. Samavati. Sketch-based dance choreography. In *Cyberworlds (CW), 2014 International Conference on*, pages 253–260, Oct 2014. doi: 10.1109/CW.2014.42.

Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3):263–284, 2007. ISSN 0920-5691. doi: 10.1007/s11263-006-9967-1. URL http://dx.doi.org/10.1007/s11263-006-9967-1.

K. Muller, A. Smolic, K. Dix, P. Kauff, and T. Wiegand. Reliability-based generation and view synthesis in layered depth video. In *Multimedia Signal Processing, 2008 IEEE 10th Workshop on*, pages 34–39, Oct 2008. doi: 10.1109/MMSP.2008.4665045.

Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

KT Ng, SC Chan, Q Wu, and HY Shum. Object-based coding for plenoptic videos. *IEEE Transactions on Circuits and Systems for Video Technology*, art. 2-s2.0-77951122418, 2010. ISSN 1051-8215. doi: 10.1109/TCSVT.2010.2041820. URL http://hdl.handle.net/10722/128744.

Anh Nguyen and Bac Le. 3d point cloud segmentation: A survey. In *Robotics, Automation and Mechatronics (RAM), 2013 6th IEEE Conference on*, pages 225–230. IEEE, 2013.

Jifeng Ning, Lei Zhang, David Zhang, and Chengke Wu. Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition*, 43(2):445 – 456, 2010. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2009.03.004. Interactive Imaging and Vision.

S.J. Norman. Dancing on occam's razor: Expressive movement and place. In C. Fernandes, editor, *Multimodal Communication in Language, Performance and Digital Media*, pages 82–94. Cambridge Scholars Publishing, Newcastle upon Tyne, 2016.

M. Okutomi and T. Kanade. A multiple-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(4):353–363, Apr 1993. ISSN 0162-8828. doi: 10.1109/34.206955.

S. Orts-Escolano, C. Rhemann, S. Fanello, W. Chang, A/ Kowdle, Y. Degtyarev, D. Kim, P. L. Davidson, S. Khamis, M. Dou, V. Tankovich, C. Loop, Q. Cai, P. A. Chou, S. Mennicken, J. Valentin, V. Pradeep, S. Wang, S. B. Kang, P. Kohli, Y. Lutchyn, C. Keskin, and S. Izadi. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 741–754, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4189-9. doi: 10.1145/2984511.2984517. URL `http://doi.acm.org/10.1145/2984511.2984517`.

R. Pajarola, M. Sainz, and P. Guidotti. Confetti: object-space point blending and splatting. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):598–608, Sept 2004. ISSN 1077-2626. doi: 10.1109/TVCG.2004.19.

Nikhil R Pal and Sankar K Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277 – 1294, 1993. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/0031-3203(93)90135-J. URL `http://www.sciencedirect.com/science/article/pii/003132039390135J`.

Mark Pauly, Richard Keiser, and Markus Gross. Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 22(3):281–289, 2003. ISSN 1467-8659. doi: 10.1111/1467-8659.00675. URL `http://dx.doi.org/10.1111/1467-8659.00675`.

Jobin T Philip, Binoshi Samuvel, K Pradeesh, and NK Nimmi. A comparative study of block matching and optical flow motion estimation algorithms. In *Emerging Research Areas: Magnetics, Machines and Drives (AICERA/iCMMD), 2014 Annual International Conference on*, pages 1–6. IEEE, 2014. doi: 10.1109/AICERA.2014.6908204.

Dilip K Prasad. Survey of the problem of object detection in real images. *International Journal of Image Processing (IJIP)*, 6(6):441, 2012.

Reinhold Preiner, Stefan Jeschke, and Michael Wimmer. Auto splats: Dynamic point cloud visualization on the gpu. In *EGPGV*, pages 139–148, 2012.

R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *Image Processing, IEEE Transactions on*, 14(3):294–307, 2005. ISSN 1057-7149. doi: 10.1109/TIP.2004.838698.

C. Rao, A. Gritai, M. Shah, and T. Syeda-Mahmood. View-invariant alignment and matching of video sequences. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 939–945 vol.2, Oct 2003. doi: 10.1109/ICCV.2003.1238449.

Liu Ren, Hanspeter Pfister, and Matthias Zwicker. Object space ewa surface splatting:

A hardware accelerated approach to high quality point rendering. *Computer Graphics Forum*, 21(3):461–470, 2002. ISSN 1467-8659. doi: 10.1111/1467-8659.00606.

Claudia Ribeiro, Rafael Kuffner, Carla Fernandes, and João Pereira. 3d annotation in contemporary dance: Enhancing the creation-tool video annotator. In *Proceedings of the 3rd International Symposium on Movement and Computing*, MOCO '16, pages 41:1–41:4, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4307-7. doi: 10.1145/2948910.2948961. URL `http://doi.acm.org/10.1145/2948910.2948961`.

Claudia Ribeiro, Rafael Kuffner dos Anjos, and Carla Fernandes. Capturing and documenting creative processes in contemporary dance. In *Proceedings of the 4th International Conference on Movement Computing*, MOCO '17, pages 7:1–7:7, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5209-3. doi: 10.1145/3077981.3078041. URL `http://doi.acm.org/10.1145/3077981.3078041`.

Iain E Richardson. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. John Wiley & Sons, 2004.

Paul L Rosin and Efstathios Ioannidis. Evaluation of global image thresholding for change detection. *Pattern Recognition Letters*, 24(14):2345–2356, 2003.

Adam Runions, Faramarz Samavati, and Przemyslaw Prusinkiewicz. Ribbons. *The Visual Computer*, 23(9):945–954, 2007. ISSN 1432-2315. doi: 10.1007/s00371-007-0153-4.

Szymon Rusinkiewicz and Marc Levoy. Qsplat: A multiresolution point rendering system for large meshes. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 343–352, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344940.

Radu Bogdan Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, October 2009.

Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011a.

Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *Robotics and automation (ICRA), 2011 IEEE International Conference on*, pages 1–4. IEEE, 2011b.

Miguel Sainz and Renato Pajarola. Point-based rendering techniques. *Computers & Graphics*, 28(6):869–879, 2004.

Maria V Sanchez-Vives and Mel Slater. From presence to consciousness through virtual reality. *Nat Rev Neurosci*, 6(4):332–339, 2005.

Johannes Schmid, Martin Sebastian Senn, Markus Gross, and Robert W. Sumner. Overcoat: An implicit canvas for 3d painting. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 28:1–28:10, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0943-1. doi: 10.1145/1964921.1964923.

Arno Schödl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video textures. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 489–498, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.345012. URL `http://dx.doi.org/10.1145/344779.345012`.

Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th International Conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-702-5. doi: 10.1145/1291233.1291311. URL `http://doi.acm.org/10.1145/1291233.1291311`.

S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 519–528, June 2006. doi: 10.1109/CVPR.2006.19.

Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 231–242, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. doi: 10.1145/280814.280882. URL `http://doi.acm.org/10.1145/280814.280882`.

Craig Shakarji. Least squares fitting algorithms of the nist algorithm testing system. *Journal of Research of the National Institute of Standards and Technology*, 103(6):633–641, November-December 1998. doi: 10.6028/jres.103.043.

Michio Shiraishi and Yasushi Yamaguchi. An algorithm for automatic painterly rendering based on local source image approximation. In *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering*, NPAR '00, pages 53–58, New York, NY, USA, 2000. ACM. ISBN 1-58113-277-8. doi: 10.1145/340916.340923.

Harry Shum and Sing Bing Kang. Review of image-based rendering techniques. In *VCIP*, pages 2–13, 2000.

S.N. Sinha and M. Pollefeys. Synchronization and calibration of camera networks from silhouettes. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 1, pages 116–119 Vol.1, Aug 2004. doi: 10.1109/ICPR.2004.1334021.

M. Slomp, H. Kawasaki, R. Furukawa, and R. Sagawa. Temporal octrees for compressing dynamic point cloud streams. In *2014 2nd International Conference on 3D Vision*, volume 2, pages 49–56, Dec 2014. doi: 10.1109/3DV.2014.79.

A. Smolic, K. Muller, K. Dix, P. Merkle, P. Kauff, and T. Wiegand. Intermediate view interpolation based on multiview video plus depth for advanced 3d video systems. In *2008 15th IEEE International Conference on Image Processing*, pages 2448–2451, 2008. doi: 10.1109/ICIP.2008.4712288.

A. Smolic, K. Mueller, P. Merkle, P. Kauff, and T. Wiegand. An overview of available and emerging 3d video formats and depth enhanced stereo as efficient generic solution. In *Picture Coding Symposium, 2009. PCS 2009*, pages 1–4, May 2009. doi: 10.1109/PCS.2009.5167358.

Maurício Sousa, Daniel Mendes, Rafael Kuffner Dos Anjos, Daniel Medeiros, Alfredo Ferreira, Alberto Raposo, João Madeiras Pereira, and Joaquim Jorge. Creepy tracker toolkit for context-aware interfaces. In *Proceedings of the Interactive Surfaces and Spaces*, ISS '17, pages 191–200, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4691-7. doi: 10.1145/3132272.3134113. URL `http://doi.acm.org/10.1145/3132272.3134113`.

Carsten Stoll, Juergen Gall, Edilson de Aguiar, Sebastian Thrun, and Christian Theobalt. Video-based reconstruction of animatable human characters. *ACM Trans. Graph.*, 29 (6):139:1–139:10, December 2010. ISSN 0730-0301. doi: 10.1145/1882261.1866161. URL `http://doi.acm.org/10.1145/1882261.1866161`.

E. Stoykova, A.A. Alatan, P. Benzie, N. Grammalidis, S. Malassiotis, J. Ostermann, S. Piekh, V. Sainov, C. Theobalt, T. Thevar, and X. Zabulis. 3-d time-varying scene capture technologies, a survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(11):1568–1586, 2007. ISSN 1051-8215. doi: 10.1109/TCSVT.2007.909975.

P.F. Sturm and S.J. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, pages –437 Vol. 1, 1999. doi: 10.1109/CVPR.1999.786974.

A Swaminathan, N Kumar, and M Ramesh Kumar. A review of numerous facial recognition techniques in image processing. *International Journal of Computer Science and MObile Computing*, 3(1):233–243, 2014.

Richard Szeliski. Video-based rendering. In *2nd European Conference on Visual Media Production*, pages 1–8, The Institution of Electrical Engineers, Savoy Place, London, UK, November 2005.

Y. Taguchi, K. Takahashi, and T. Naemura. Real-time all-in-focus video-based rendering

using a network camera array. In *2008 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 241–244, May 2008. doi: 10.1109/3DTV.2008.4547853.

Guangming Tang, Dawei Wei, and Jia Hu. A fast and stable bold feature extraction algorithm. In *Instrumentation and Measurement, Computer, Communication and Control (IMCCC), 2014 Fourth International Conference on*, pages 799–804, Sept 2014. doi: 10.1109/IMCCC.2014.169.

Masayuki Tanimoto. Ftv: Free-viewpoint television. *Signal Processing: Image Communication*, 27(6):555 – 570, 2012. ISSN 0923-5965. doi: http://dx.doi.org/10.1016/j.image.2012.02.016. URL v.

Manuel Veit and Antonio Capobianco. Go'then'tag: A 3-d point cloud annotation technique. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 193–194. IEEE, 2014.

George Vogiatzis and Carlos Hernández. Video-based, real-time multi-view stereo. *Image and Vision Computing*, 29(7):434 – 441, 2011. ISSN 0262-8856. doi: 10.1016/j.imavis.2011.01.006.

Marco Volino and Adrian Hilton. Layered view-dependent texture maps. In *Proceedings of the 10th European Conference on Visual Media Production*, CVMP '13, pages 16:1–16:8, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2589-9. doi: 10.1145/2534008.2534022. URL http://doi.acm.org/10.1145/2534008.2534022.

H.-H. Vu, P. Labatut, J.-P. Pons, and R. Keriven. High accuracy and visibility-consistent dense multiview stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(5):889–901, May 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.172.

C. Wang, S. C. Chan, C. H. Ho, A. L. Liu, and H. Y. Shum. A real-time image-based rendering and compression system with kinect depth camera. In *2014 19th International Conference on Digital Signal Processing*, pages 626–630, Aug 2014a. doi: 10.1109/ICDSP.2014.6900740.

Oliver Wang, Christopher Schroers, Henning Zimmer, Markus Gross, and Alexander Sorkine-Hornung. Videosnapping: interactive synchronization of multiple videos. *ACM Transactions on Graphics (TOG)*, 33(4):77, 2014b.

Oliver Wasenmüller and Didier Stricker. *Comparison of Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision*, pages 34–45. Springer International Publishing, Cham, 2017. ISBN 978-3-319-54427-4. doi: 10.1007/978-3-319-54427-4_3. URL http://dx.doi.org/10.1007/978-3-319-54427-4_3.

Lee Alan Westover. *Splatting: a parallel, feed-forward volume rendering algorithm*. PhD thesis, University of North Carolina at Chapel Hill, 1991.

Sarah Whatley. Siobhan davies replay:(re) visiting the digital archive. *International Journal of Performance Arts and Digital Media*, 9(1):83–98, 2013.

Andrew D. Wilson. Fast lossless depth image compression. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*, ISS '17, pages 100–105, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4691-7. doi: 10.1145/3132272.3134144. URL `http://doi.acm.org/10.1145/3132272.3134144`.

Jason Wither, Stephen DiVerdi, and Tobias Höllerer. Annotation in outdoor augmented reality. *Computers & Graphics*, 33(6):679–689, 2009.

Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. Elan: a professional framework for multimodality research. In *Proceedings of LREC*, volume 2006, page 5th, 2006.

Chenglei Wu, Carsten Stoll, Levi Valgaerts, and Christian Theobalt. On-set performance capture of multiple actors with a stereo camera. *ACM Trans. Graph.*, 32(6):161:1–161:11, November 2013. ISSN 0730-0301. doi: 10.1145/2508363.2508418. URL `http://doi.acm.org/10.1145/2508363.2508418`.

Hui Xu and Baoquan Chen. Stylized rendering of 3d scanned real world environments. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering*, NPAR '04, pages 25–34, New York, NY, USA, 2004. ACM. ISBN 1-58113-887-3. doi: 10.1145/987657.987662.

Hui Xu, Nathan Gossett, and Baoquan Chen. Pointworks: Abstraction and rendering of sparsely scanned outdoor environments. In Alexander Keller and Henrik Wann Jensen, editors, *Eurographics Workshop on Rendering*. The Eurographics Association, 2004. ISBN 3-905673-12-6. doi: 10.2312/EGWR/EGSR04/045-052.

Li Xu, Jiaya Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(9):1744–1757, Sept 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.236.

Razali Yaakob, Alihossein Aryanfar, Alfian Abdul Halin, and Nasir Sulaiman. A comparison of different block matching algorithms for motion estimation. *Procedia Technology*, 11(0):199 – 205, 2013. ISSN 2212-0173. doi: http://dx.doi.org/10.1016/j.protcy.2013.12.181. URL `http://www.sciencedirect.com/science/article/pii/S2212017313003356`. 4th International Conference on Electrical Engineering and Informatics, {ICEEI} 2013.

Chuan-Kai Yang and Hui-Lin Yang. Realization of seurat's pointillism via non-photorealistic rendering. *The Visual Computer*, 24(5):303–322, 2008. doi: 10.1007/s00371-007-0183-y.

Xiaohui Yang, Ju Liu, Jiande Sun, Xinchao Li, Wei Liu, and Yuling Gao. Dibr based

view synthesis for free-viewpoint television. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), 2011*, pages 1–4, May 2011. doi: 10.1109/3DTV.2011.5877165.

Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), December 2006. ISSN 0360-0300. doi: 10.1145/1177352.1177355. URL `http://doi.acm.org/10.1145/1177352.1177355`.

Seung-Uk Yoon, Eun-Kyung Lee, Sung-Yeol Kim, and Yo-Sung Ho. A framework for multi-view video coding using layered depth images. In *Advances in Multimedia Information Processing-PCM 2005*, pages 431–442. Springer, 2005.

Seung-Uk Yoon, Eun-Kyung Lee, Sung-Yeol Kim, and Yo-Sung Ho. A framework for representation and processing of multi-view video using the concept of layered depth image. *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, 46(2-3):87–102, 2007.

Ming Zeng, Fukai Zhao, Jiaxiang Zheng, and Xinguo Liu. Octree-based fusion for realtime 3d reconstruction. *Graphical Models*, 75(3):126 – 136, 2013. ISSN 1524-0703. doi: http://dx.doi.org/10.1016/j.gmod.2012.09.002. Computational Visual Media Conference 2012.

Qingming Zhan, Yubin Liang, and Yinghui Xiao. Color-based segmentation of pointclouds. *Laserscanning '09*, 38(Part 3/W8):248–252, Sept. 2009.

Cha Zhang. Multiview imaging and 3dtv. *IEEE Signal Processing Magazine*, 1053 (5888/07), 2007.

Cha Zhang and Tsuhan Chen. A survey on image-based rendering & representation, sampling and compression. *Signal Processing: Image Communication*, 19(1):1 – 28, 2004. ISSN 0923-5965. doi: 10.1016/j.image.2003.07.001.

Shan Zhu and Kai-Kuang Ma. A new diamond search algorithm for fast block-matching motion estimation. *Image Processing, IEEE Transactions on*, 9(2):287–290, Feb 2000. ISSN 1057-7149. doi: 10.1109/83.821744.

C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 600–608, New York, NY, USA, 2004. ACM. doi: 10.1145/1186562.1015766. URL `http://doi.acm.org/10.1145/1186562.1015766`.

Matthias Zwicker, Jussi Räsänen, Mario Botsch, Carsten Dachsbacher, and Mark Pauly. Perspective accurate splatting, 2004.