Generating and Streaming Immersive Sports Video Content

by

Kiana Calagari

M.Sc., Sharif University of Technology, Tehran, Iran, 2012
B.Sc., Sharif University of Technology, Tehran, Iran, 2010

Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Science

© Kiana Calagari 2017
SIMON FRASER UNIVERSITY
Summer 2017

All rights reserved.
However, in accordance with the Copyright Act of Canada, this work may be reproduced without authorization under the conditions for “Fair Dealing.” Therefore, limited reproduction of this work for the purposes of private study, research, education, satire, parody, criticism, review and news reporting is likely to be in accordance with the law, particularly if cited appropriately.
Approval

Name: Kiana Calagari
Degree: Doctor of Philosophy (Computing Science)
Title: Generating and Streaming Immersive Sports Video Content
Examination Committee:

Mohamed Hefeeda  
Senior Supervisor  
Professor

Shervin Shirmohammadi  
Supervisor  
Professor

Ze-Nian Li  
Internal Examiner  
Professor  
School of Computing Science

Carsten Gruwedz  
External Examiner  
Professor  
Simula Research Laboratory

Chair: Ryan Shea  
Research Associate

Date Defended: 18 August 2017
Abstract

Stereoscopic 3D videos have already become popular in movie theaters with most productions being released in this format. More recently, with the availability of commodity Virtual Reality (VR) products, immersive video content is receiving even more interest. A wide spread adoption of immersive devices and displays is hindered by the lack of content that matches the user expectations. Producing immersive videos is far more costly and time-consuming than regular 2D videos, which makes it challenging and thus rarely attempted, especially for live events, such as sports games. In addition, immersive content needs to be adapted for viewing on different displays/devices. To address these challenges, we first propose a new system for 3D video streaming that provides automatic depth adjustments as one of its key features. Our system takes into account both the content and the display type in order to customize 3D videos and optimize the viewing experience. Our stereoscopic video streaming system was implemented, deployed and tested with real users. Results show that between 60% to 70% of the shots can benefit from our system and more than 25% depth enhancement can be achieved. Next, we propose a novel, data-driven method that converts 2D videos to 3D by transferring depth information from a database of similar 3D videos. Our method then reconstructs the depth map while ensuring temporal coherency using a spatio-temporal formulation of Poisson reconstruction. Results show that our method produces high-quality 3D videos that are almost indistinguishable from videos shot by stereo cameras, while achieving up to 20% improvement in the perceived depth compared to the current state-of-the-art method. Furthermore, we extend our work in the direction of VR, and propose using video feeds from regular broadcasting cameras to generate sports VR content. We generate a wide-angle panorama by utilizing the motion of the main camera. We then use various techniques to remove the parallax, align all video feeds, and overlay them on the panorama. Subjective studies show that our generated content provides an immersive experience similar to ground-truth content captured using a 360 camera, with most subjects rating their sense of presence from Good to Excellent.

Keywords: Immersive content, 3D video, Depth optimization, Stereoscopic retargeting, Depth personalization, 2D-to-3D conversion, Data-driven, Depth estimation, Virtual reality, 2D-to-VR conversion, Multiple cameras, Sports videos.
To my beloved Tarek.
Acknowledgements

I would like to express my special thanks and sincere gratitude to my senior supervisor, Prof. Mohamed Hefeeda. This thesis wouldn’t have been possible without his valuable guidance, continuous encouragement, and genuine care and support. I would also like to extend my thanks to my supervisor, Prof. Shervin Shirmohammadi, whom I wouldn’t have been here in the first place, if it wasn’t for him.

I wish to express my sincere thanks to my examiners, Prof. Ze-Nian Li and Prof. Carsten Gruwodz, who have put the time and effort to review this thesis carefully and have improved it with their thoughtful and constructive comments. Also, many thanks to Dr. Ryan Shea for taking the time to chair my thesis defence.

I would like to thank all my friends, near or far, who have been there for me whenever I needed their support. I also would like to thank all my colleagues at the Network Systems Lab who have made the lab a pleasant environment to work in. In addition, I am grateful for the help of all collaborators, co-authors, and colleagues who facilitated the advancing of my work.

And last but definitely not least, I wish to express my utmost gratitude to my beloved husband and family for their endless love, patience and support throughout the years.
# Table of Contents

- Approval ii
- Abstract iii
- Dedication iv
- Acknowledgements iv
- Table of Contents vi
- List of Tables ix
- List of Figures x

## 1 Introduction

1.1 Overview ......................................................... 1
1.2 Thesis Contributions ........................................... 2
   1.2.1 Customization and Streaming of Stereoscopic Sports Videos ............ 2
   1.2.2 Gradient-based 2D-to-3D Conversion for Sports Videos ................... 3
   1.2.3 VR Content Generation from Regular Camera Feeds for Sports .......... 4
1.3 Thesis Organization ........................................... 5

## 2 Background

2.1 Introduction .................................................... 6
2.2 3D Videos ..................................................... 7
   2.2.1 3D Perception ................................................ 7
   2.2.2 3D Viewing Technologies .................................. 8
   2.2.3 3D Video Representations ................................ 11
   2.2.4 Principles for Producing Good-quality 3D Content ..................... 13
2.3 VR Content Representations .................................. 14
   2.3.1 Equirectangular Projection .................................. 14
   2.3.2 Cubemap Projection ......................................... 15
   2.3.3 Pyramid Projection ......................................... 15
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6.3 Objective Experiments</td>
<td>67</td>
</tr>
<tr>
<td>4.6.4 Computational Complexity</td>
<td>71</td>
</tr>
<tr>
<td>4.7 Summary</td>
<td>71</td>
</tr>
<tr>
<td>5 VR Content Generation from Regular Camera Feeds for Sports</td>
<td>72</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>72</td>
</tr>
<tr>
<td>5.2 Related Work</td>
<td>73</td>
</tr>
<tr>
<td>5.3 Proposed Method</td>
<td>75</td>
</tr>
<tr>
<td>5.3.1 Overview</td>
<td>75</td>
</tr>
<tr>
<td>5.3.2 Generating Static Panorama</td>
<td>76</td>
</tr>
<tr>
<td>5.3.3 Removing Parallax</td>
<td>78</td>
</tr>
<tr>
<td>5.3.4 Overlaying Using Poisson Blending</td>
<td>80</td>
</tr>
<tr>
<td>5.4 Evaluation</td>
<td>82</td>
</tr>
<tr>
<td>5.4.1 Setup</td>
<td>82</td>
</tr>
<tr>
<td>5.4.2 Evaluation of our Technique</td>
<td>83</td>
</tr>
<tr>
<td>5.4.3 Comparison against Original 360 Content</td>
<td>84</td>
</tr>
<tr>
<td>5.4.4 Analysis of Player Misplacements</td>
<td>84</td>
</tr>
<tr>
<td>5.5 Summary</td>
<td>85</td>
</tr>
<tr>
<td>6 Conclusions and Future Work</td>
<td>87</td>
</tr>
<tr>
<td>6.1 Conclusions</td>
<td>87</td>
</tr>
<tr>
<td>6.2 Future Work</td>
<td>89</td>
</tr>
<tr>
<td>Bibliography</td>
<td>91</td>
</tr>
</tbody>
</table>
List of Tables

Table 3.1  Shot analysis of the Milan vs. Barcelona full 3D soccer game on 20/2/2013 in the UEFA Champions League. ............................................. 42
Table 3.2  Shot analysis of the Manchester United vs. Liverpool full 3D soccer game on 16/3/2014 in the English Premiere League. ............................................. 43
Table 3.3  Shot analysis of the R. Nadal vs. N. Djokovic 3D tennis clip on 3/7/2011 in the Wimbledon competition. ............................................. 43
List of Figures

Figure 2.1 Simple stereo camera model where the camera axes are in parallel. 8
Figure 2.2 Disparity (d) vs. perceived depth (z) when viewing a 3D display. p is the viewing distance, and e is the inter-ocular distance [30]. 9
Figure 2.3 In the autostereoscopic (glasses free) technologies, corresponding light waves are redirected to each eye by the use of (a) parallax barriers or (b) lenticular lens [10]. 10
Figure 2.4 Video plus depth representation of 3D video [91]. 12
Figure 2.5 Three intermediate views synthesized using two reference views and their depth maps [58]. 13
Figure 2.6 Demonstration of the equirectangular projection. The rectangle on the left represents the equirectangular projection of the sphere on the right. 15
Figure 2.7 Examples of images that can benefit from (a) light and shading, (b) focus/defocus, and (c) linear perspective depth cues. 19
Figure 2.8 Block diagram of the overall algorithm for 2D-to-3D conversion in [74]. 21
Figure 2.9 Block diagram of the overall algorithm for 2D-to-3D conversion in [71]. 22
Figure 3.1 Design of the proposed 3D streaming system, which offers depth enhancement for 3D videos viewed on different displays. 26
Figure 3.2 Example of the slant operation. A parallelogram-shaped region is mapped back onto the whole image area, modifying the vertical component $g_y$ of the depth gradient $g$. 29
Figure 3.3 Example of the stretch operation. The left and right views are horizontally scaled. In effect, the horizontal component $g_x$ of the depth gradient $g$ is changed. 30
Figure 3.4 Examples of shots in a soccer video that have planar depth structure. The depth maps were determined using optical flow estimation methods [119, 24], and were further enhanced by cross bilateral filtering [93]. Note, that they are provided for visualization purposes, and our method does not require computationally expensive estimation of accurate, dense stereo correspondences. 31
Figure 3.5 Our method has increased the disparity range while preserving the depth structure and orientation (the top images are the original disparity maps, while the bottom ones show depth after our optimization).

Figure 3.6 Left: ROC curve for our binary shot classifier. Right: Four examples of shots with corresponding $R^2$ values. Top two have low $R^2$ value, therefore, their depth will be unchanged. On the other hand, bottom two are long shots and due to the high $R^2$ value (above the threshold) their depth will be modified.

Figure 3.7 Our system can produce personalized depth, based on viewer’s preferences.

Figure 3.8 Our system provides users with three options for controlling the popping out effect: (a) Increasing pop out effect, (b) Increasing depth range while not increasing the pop out effect, and (c) Removing pop out effect.

Figure 3.9 Mean expansion value and standard error of the mean for two displays.

Figure 3.10 Histogram of the expansion values preferred by subjects.

Figure 3.11 Difference Mean Opinion Score (DMOS) between different depth levels and the original 3D sequence, showing the impact of each level on depth perception and visual comfort. Zero implies that the sequence is the same as the original. Error bars represent the standard error of the mean. Higher levels translate to higher depth values. Level 0 is the original video.

Figure 3.12 Difference Mean Opinion Score (DMOS) between our optimized version and the original 3D sequence. Our method improves depth quality for all tested videos on all displays. Error bars represent the standard error of the mean.

Figure 3.13 Time required to process each operation in our system. The horizontal line is at $t = 42$ msec, representing real time processing at 24 fps.

Figure 4.1 The proposed 2D-to-3D conversion method.

Figure 4.2 The main components of the data-driven depth estimation method.

Figure 4.3 The effect of different steps in the proposed depth estimation method: (a) Query, (b) Subset of $K$ matching candidates, (c) Created matched image, (d) Object boundary cuts, (e) Depth estimation using Poisson reconstruction, (f) Gradient refinement and Poisson reconstruction, (g) Depth with object boundary cuts, (h) Final depth estimation with smoothness, and (i) A zoomed and amplified version of the yellow block in $h$.

Figure 4.4 Construction of matrix $A$ of the Poisson equation. (a) An example of $4 \times 4$ image showing a sample pixel $p$ and its neighbors. (b) The coefficients of Eq. (4.3) for pixel $p$. (c) The non-zero values in $A$ for the row corresponding to pixel $p$.

Figure 4.5 The refinement factor of $G_x$ for $\alpha = 60$.
Figure 4.6  Shot classification: An example of the detected field area, and the area covered by the fitted convex hull, for a close-up (top row) and non close-up (bottom row) frame. Red lines show the boundaries of the fitted convex hulls. .......................... 57

Figure 4.7  An example of object mask creation for non close-ups. For a more complete detection of players, the final object mask is generated as a fusion of the non-field detection and the background subtraction approach. For instance, players in the yellow box are missing from the background subtraction approach, while the players in the green boxes are small from the field only method. However the final mask recovers the missing players and generates a more-complete object mask. .......................... 58

Figure 4.8  For a close-up shot, a combination of feature point trajectories [87] and frame-to-frame motion [81] is used to generate background and foreground matting strokes. The method in [78] is then used to extract a dense players matte. Finally, field segmentation removes matting inaccuracies. ........... 59

Figure 4.9  Depth estimation for different types of shots using our method. Our method handles a wide variety of shots including Close-ups (e.g., top, left-most), Medium Shots (e.g., top, right-most), Bird’s Eye View (e.g., bottom, left-most) and Long Shots (e.g., bottom, right-most). ......................... 60

Figure 4.10  Mean opinion scores of depth perception and visual comfort for different types of soccer scenes. .......................... 63

Figure 4.11  Mean opinion scores of depth perception and visual comfort for different non-soccer field sports. .......................... 63

Figure 4.12  Difference mean opinion score (DMOS) between our converted sequences and the original 3D. Zero implies that our converted sequence is the same as the original 3D. .......................... 64

Figure 4.13  Difference mean opinion score (DMOS) between our converted sequences and Depth Transfer DT+. Positive DMOS means that our technique is preferred over DT+. .......................... 64

Figure 4.14  Depth estimation for different sequences using: DT, DT+ and our method DGC. DT generates erroneous estimates, DT+ generates noisy measurements and does not detect players. Our technique outperforms both approaches. .......................... 65

Figure 4.15  Mean opinion scores of depth perception and visual comfort for two sequences of different temporal complexity, where three methods are compared: our spatio-temporal Poisson reconstruction, temporal smoothness as a post-process, and without temporal smoothness. .......................... 66
Figure 4.16 Top row: Frame 3 of a synthetic sequence. Bottom row: Frame 24 of a real sequence. We show the depth extracted using: Ground-truth/Stereo Correspondence [24], DT, DT+ and DGC. Our technique DGC best reassembles the Ground-truth/Stereo Correspondence in both sequences.

Figure 4.17 An objective comparison between our DGC method and the closest method in the literature DT, and its extension DT+ on a (a) synthetic soccer sequence and (b) real soccer sequence. Our method generates much lower MAE than both DT and DT+ for both sequences.

Figure 4.18 The average depth values for each frame of a (a) synthetic sequence and (b) real sequence when using different temporal window sizes. While without temporal smoothness the scene experiences sudden changes of depth, the depth changes are much smoother when temporal smoothness is applied.

Figure 4.19 Mean Absolute Error (MAE) between each two consecutive frames of a (a) synthetic sequence and (b) real sequence, when using different temporal window sizes.

Figure 5.1 The 3 stages of our immersive content generation technique, and their main components.

Figure 5.2 Example of a static panorama generated from a basketball game.

Figure 5.3 Sample frames of 3 different video feeds positioned around the field. All frames are shot at the same time instant.

Figure 5.4 Removing parallax between the right camera frame in Fig.5.3(c) and the frame from the main camera in Fig.5.3(b) using its object mask. The object mask is generated using the background subtraction technique.

Figure 5.5 Overlaying the main feed on the static panorama, and copying the missing players from the left and right feeds. The blue arrows indicate that the players are copied from the left or right feeds.

Figure 5.6 Examples of final panoramas generated by our technique for different games: basketball (top), hockey (middle), and volleyball (bottom). The blue arrows indicate the players that have been copied from the left or right feeds.

Figure 5.7 Mean opinion scores of video quality and sense of presence for different games.

Figure 5.8 Difference mean opinion score (DMOS) between content generated using our technique and original 360 content captured using 360 camera. A value of zero implies that the results from our technique are the same as the original 360.

Figure 5.9 Average displacement of copied players for different games. Error bars represent the standard deviation.
Chapter 1

Introduction

1.1 Overview

Stereoscopic 3D content has already become mainstream in movie theaters. Many productions are released in stereoscopic format and there is a significant audience that enjoys this more immersive experience [65, 49]. The growth of 3D display technologies and high bandwidth Internet encouraged many on-line streaming services, e.g., YouTube [15] and Trivido [14], to support 3D streaming. Moreover, the wide availability of various stereoscopic 3D displays from 3D-ready TV-sets to mobile devices, largely facilitated the popularity of 3D videos at home. However, a widespread adoption of 3D displays is hindered by the lack of high-quality 3D content. Currently, 3D videos are not widely available, since producing 3D videos is far more challenging than regular 2D videos. One promising solution to address this need is to use automated 2D-to-3D conversion. High quality 2D-to-3D conversion methods not only can make it possible to view old productions in 3D, but can also facilitate the production of new 3D videos.

In addition, while most stereoscopic 3D content is designed for viewing in movie theaters, where viewing conditions do not vary significantly, rendering the same content for viewing on home displays requires additional adjustments. Both 3D perception and visual comfort highly depend on viewing conditions (e.g., display size and viewing distance) [35, 115, 117, 107]. Also, there is a large variety of different 3D display types, each requiring a different video input (e.g., side-by-side, temporal interleaving, and spatial interleaving) with different capabilities and limitations [21]. Therefore, further content customization is required based on the viewing conditions. Moreover, stereoscopic content should be customized to account for differences in user preferences. A content that provides a good experience for one user does not necessarily provide a good experience for others. As a result, providing the same content for all displays and users results in sub-optimal quality, both in terms of depth perception and visual comfort.

To address these challenges, we first propose a new system for 3D sports video streaming that improves viewing experience by adjusting depth based on viewing conditions and display technology. Our system includes a depth personalization method which allows users to adjust the amount of
depth according to their preferences. Next, we propose a novel, data-driven spatio-temporal method that generates 3D videos by transferring depth information from a database of similar 3D videos. While a database of 3D videos with accurate depth is very difficult to obtain, our key observation is to leverage high-quality computer generated content in current computer games to generate a reference data set. In our experiments, we created a reference data set using the FIFA 13 computer game and used it to convert 2D soccer videos to 3D. In addition, our proposed spatio-temporal formulation can achieve temporally smooth results while being parallelizable.

Furthermore, we extend our work in the direction of Virtual reality (VR), which is experiencing a growing interest in the multimedia industry. Similar to 3D videos, one of the main problems that prevents VR from being adopted on a wider scale is the lack of content. The majority of current VR content is synthetic, mostly for the gaming industry. For generating real VR content (such as sports games), the only approach is by using 360 camera rigs. Such rigs contain multiple cameras stacked next to each other in a way that maximizes the field of view [4, 19, 13]. Camera outputs are then stitched together to enhance the overall sense of immersion. Current solutions for capturing high-quality VR content require upgrading the entire production pipeline. This includes data capturing, processing and distribution. Such upgrade is expensive to set-up and operate, which makes it an unappealing solution. We propose an alternative approach for VR content generation that converts the traditional broadcast camera feeds to VR through a post-processing stage. Producing VR content from the traditional video feeds is a complex task and requires addressing multiple challenges. For example, due to the limited coverage of the main camera, players tend to appear/disappear through out the recording. Such effect significantly degrades the feeling of immersion. To overcome this problem, we identify and retrieve the missing players from different camera feeds.

1.2 Thesis Contributions

1.2.1 Customization and Streaming of Stereoscopic Sports Videos

Streaming 3D videos to a variety of different 3D displays is challenging. This is because first, there are different 3D display technologies, each requiring a specific form of input. Second, there are different 3D display sizes, each having a different viewing condition. Thus, for a correct depth reproduction and a comfortable viewing experience on a particular device additional post-processing adjustments are required to account for the viewing distance and/or the screen size [115, 117, 107]. In addition, not all users prefer the same amount of depth. Therefore, viewer’s preferences should also be taken into account. Previous works on 3D streaming have addressed various problems including content representation, content acquisition, encoding, and transmission of both stereoscopic and multiview content. However, we are not aware of any 3D streaming system that accounts for display technologies and adaptively customizes the depth based on the viewing distance, display size, and viewer’s preferences.
We address this problem by presenting a new system for streaming stereoscopic content. Its key feature is a computationally efficient depth adjustment technique which can automatically optimize viewing experience of sports videos. Additionally, the method enables depth personalization to allow users to adjust the amount of depth according to their preferences. Our proposed depth adjustment method is well-suited for videos of field sports such as soccer, hockey, and tennis. Our method is computationally efficient and it does not introduce any visual artifacts. We have implemented our 3D streaming system and conducted user studies that show the benefits of content adaption. Our contributions in this work can be summarized as follows [30, 26]:

• We propose a novel method for depth expansion and compression for stereoscopic 3D videos. Our method performs simple image processing operations, does not require creating accurate depth maps, and does not introduce visual artifacts. The main target application for our method is sport videos e.g., soccer, hockey, and tennis. In such applications, preserving the scene structure, e.g., the straightness of the lines, is extremely important. Our method preserves the scene structure.

• We propose a method for depth personalization, which allows users to control the depth in 3D videos to suit their comfort level. This is an important feature as not all users prefer the same amount of depth. Some users do not like depth at all, while others are fascinated by 3D content with striking and aggressive depths.

• We conduct user studies to show the need for our system in general, and to demonstrate the achieved gain in depth enhancements by the proposed system, which can be more than 25%. In addition, we show that our method can enhance between 60% to 70% of the shots in 3D field sports.

1.2.2 Gradient-based 2D-to-3D Conversion for Sports Videos

One promising solution to address the lack of high-quality 3D content is to use automated 2D-to-3D conversion. However, current automated conversion methods ([94, 73, 106, 74, 71]) are lacking. While being general, they cannot handle complex motion ([94, 74, 71]), or produce low-quality results that exhibit artifacts such as the well-known ‘card-board effect’ where objects appear flat when viewed in stereo [73, 106]. While, data-driven methods are an alternative way of computing depth maps, current methods ([74, 71]) only produce high-quality results if there is a strong similarity between the examined frame and the database. In addition, current 3D video databases are limited and mostly shot indoors, which makes them not suitable for a data-driven 2D-to-3D conversion of sports videos.

We address this problem by showing how to construct a high-quality, domain-specific conversion method for sports videos. We propose a novel, data-driven method that relies on a reference database of 3D stereoscopic videos. In order to generate this reference database, we leverage high-quality computer generated content in current sport computer games. Once we retrieve similar 3D
video frames, our technique transfers depth gradients to the target frame while respecting object boundaries, solves for the output depth map, and then generates stereoscopic video.

One of the main challenges in the proposed 2D-to-3D conversion method is providing temporal smoothness in the depth estimation process without limiting its parallelizable feature. If the depth estimation is performed independently for each frame, the generated depth maps will not be temporally smooth and can vary significantly between subsequent frames. In order to have a robust 2D-to-3D conversion, we enforce temporal smoothness by performing a three-dimensional gradient based depth reconstruction, which uses temporal gradients in addition to the spatial gradients to reconstruct the depth.

We validate our method by conducting user-studies that evaluate depth perception and visual comfort of multiple converted 3D videos. Our contributions in this work can be summarized as follows [27, 28]:

• We use computer generated depth from current computer sport games to construct a 2D+Depth database. Since the video quality of current computer games has come close to that of real videos, our approach offers two advantages: 1) we obtain a diverse database of video frames to facilitate good matching with input video frames; 2) for each video frame, we obtain an accurate depth map with perfect depth discontinuities.

• We transfer depth gradients using a block based approach, as opposed to previous approaches that use absolute depth over the whole frame. Our approach offers multiple advantages, such as achieving a finer depth assignment and requiring a much smaller database.

• We propose a temporal smoothness method to control depth variations in successive frames, which improves the visual comfort for users.

• We conduct user studies to show that our method produces 3D videos that are almost indistinguishable from videos originally shot in stereo. The studies also show that our method significantly outperforms the current state-of-the-art method. For example, up to 20% improvement in the perceived depth is achieved by our method.

1.2.3 VR Content Generation from Regular Camera Feeds for Sports

With the recent popularity of commodity VR products, immersive video content is receiving significant interest. However, current solutions for producing high-quality VR content require upgrading the entire production pipeline. The expensive equipment and special training required for setting up and operating such solutions causes them to be very time-consuming and costly. On the other hand, existing setups for producing and broadcasting sports events in 2D are usually rich and well-equipped. Most major stadiums have many high-end cameras around the field operated by professional staff.

We propose an alternative approach for VR content generation that converts the traditional broadcast video feeds to VR content through a post-processing stage. Using our technique the same
setup used for regular broadcasting of sports events can be used to generate immersive content. To produce VR content from the traditional video feeds, first, we need to widen the field of view to at least 180 degrees. We do so by utilizing the motion of the main camera (the camera chosen as the user’s viewing position) to generate a wide-angle panorama. We then use different camera feeds to identify and retrieve the players missing in the main camera feed. However, to do so, all feeds need to be aligned with respect to a reference frame, which is challenging because of the large distance between cameras. Due to this large distance a huge amount of parallax is present between different views which causes the position of objects and the orientation of lines to appear differently when viewed from different cameras. While such large amounts of parallax cause state-of-the-art image alignment techniques, e.g. [130], to fail, we use various techniques to remove the parallax between all video feeds and successfully align them.

We evaluate our technique by conducting subjective studies that measure the sense of presence and perceived video quality. Our contributions in this work can be summarized as follows [29]:

- We increase the viewing angle by utilizing the camera rotation, and generating a panorama image which includes the static parts of the scene.
- We overlay the field and players on the panorama. We then identify and copy any missing players from other feeds.
- We propose a method for aligning the video feeds from cameras placed meters away from each other. This is challenging because of the large distance between cameras, which causes the position of objects and the orientation of lines to appear differently when viewed from different cameras.
- We conduct subjective studies that compare our generated content to ground-truth content captured using a 360 camera rig. The studies show that our generated content provides an immersive experience similar to the ground-truth content, with most subjects rating their sense of presence from Good to Excellent.

### 1.3 Thesis Organization

The chapters of this thesis are organized as follows. In Chapter 2, we present a background on 3D principles and immersive technologies, and discuss various 3D streaming systems and state-of-the-art solutions for 3D/VR customization and conversion. In Chapter 3 we propose a system for customization, streaming and personalization of 3D sports videos. In Chapter 4, we propose a data-driven gradient-based 2D-to-3D conversion method for sports videos. In Chapter 5, we propose a method for generating immersive sports content from regular broadcasting camera feeds. We conclude the thesis and discuss future research directions in Chapter 6.
Chapter 2

Background

2.1 Introduction

In this chapter we discuss the basics of capturing and perceiving a video in 3D/VR, followed by different 3D/VR viewing technologies and representations. We also describe the main principles that should be taken into account when creating 3D content. We then describe the research aspects, such as streaming 3D videos, customizing 3D videos, and 2D-to-3D conversion, and discuss their state-of-the-art solutions.

Producing high-quality stereoscopic 3D content requires significantly more effort than preparing regular video footage. For viewers to perceive the depth in a 3D video perfectly and without any discomfort such as headaches or nausea, 3D videos need to be carefully adjusted to specific viewing conditions before they are shown to viewers. Content adaptation is not a simple task since current 3D displays (e.g., TV sets, desktop displays, laptops, and mobile devices) vary in terms of technology and size. Both depth perception and visual comfort highly depend on viewing conditions (e.g., display size and viewing distance). Therefore, content customization should be based on the viewing conditions. In addition, stereoscopic content should be manipulated to account for differences in user preferences. A content that provides a good experience for one user does not necessarily provide a good experience for others. As a result, providing the same content for all displays and users results in sub-optimal quality, both in terms of depth perception and visual comfort.

Another challenge that hinders the wide spread adoption of 3D videos is the lack of high-quality 3D content. Shooting 3D videos is far more costly and time-consuming than regular 2D videos, which makes it challenging and thus rarely attempted, especially for live events, such as soccer games. A promising solution to address this problem is to use automated 2D-to-3D conversion methods. These challenges and other aspects of 3D video systems have received significant interest from both academia and industry.
2.2 3D Videos

2.2.1 3D Perception

We can sense the depth of objects around us as our visual system is able to extract a variety of depth cues from a scene. Depth cues can be classified into two categories, monocular cues and binocular cues. Monocular cues can be observed by just one eye, such as the relative size of objects, focus, linear perspective, etc. Binocular cues are based on the ability of our brain to fuse the views from both our eyes together and use it to perceive the depth. Due to the distance between our eyes these two views are slightly different. This difference is the key to 3D perception. Therefore, in order to perceive an image in 3D we need two slightly different images, one for the left eye and one for the right eye. By forcing each eye to just see its corresponding image, we can trick the brain and create the illusion of perceiving a real 3D scene. As a result, cameras used for capturing 3D videos include two built-in lenses separated from each other with a distance similar to the human eyes (about 6.5cm). Each of these lenses simulates one of the eyes and captures the view directed to the corresponding eye. As simple as it may sound, we should note that compared to normal cameras, 3D cameras are bigger, more complicated to use and more expensive. To capture a live event such as a soccer game in 3D, many 3D camera rigs are needed to cover the whole area from different angles, which can cause a significant amount of overhead.

The amount of perceived depth for each pixel is a factor of binocular disparity. Binocular disparity is the distance between the location of an object in the left and right views. For each pixel in the left view, there is a corresponding pixel in the right view which is located a few pixels away. The distance between the x-positions of two corresponding pixels is called disparity. Disparities are detected by the human visual system and interpreted as depth. The relationship between the depth \( z \) and disparity \( d \) depends on the 3D camera model being used. In the following we discuss the two main stereo camera models [79].

**Simple Stereo Camera Model**

In this case the optical axes of the left and right cameras are parallel. Eq. 2.1 shows the relationship between depth \( z \) and disparity \( d \) in this model, where \( f \) is the focal length and \( b \) is camera separation. It can be seen that disparity is inversely proportional to the depth. Thus, near objects have large disparity values, while far objects have small disparity values. When depth reaches infinity, the disparity would reach zero. Fig. 2.1, demonstrates an example where \( P(x, y, z) \) is a 3D point in space, \( x_l \) and \( x_r \) are the x-coordinates of the point’s projection on the left and right camera image planes, which their difference is the disparity, and the centre of the world coordinate system is assumed to be right in the middle of the two cameras.

\[
    z = \frac{fb}{d}. \tag{2.1}
\]
Figure 2.1: Simple stereo camera model where the camera axes are in parallel.

Toed-in Stereo Camera Model

Unlike the simple camera model, human eyes are not parallel when focusing on an object near to us. When focusing on a near object, our eyes converge, which means they rotate in opposite directions around a vertical axis in order to obtain a zero disparity on the object in focus. When viewing a 3D video on a screen, our eyes focus on the screen plane. In this case, if the disparity between the two views on the screen is zero the object appears on the screen plane. An object is perceived behind the screen plane if the disparity is positive, and in front of the screen plane if the disparity is negative. Fig. 2.2 shows the relationship between perceived depth and disparity when viewing a 3D display. This behaviour of our eyes can be modelled by toed-in stereo cameras which usually have converging camera axes instead of parallel ones.

2.2.2 3D Viewing Technologies

In order to direct each of the left and right views to the corresponding eye, different viewing technologies have been developed which can be categorized into: aided-view technologies and auto-stereoscopic technologies.

Aided-view Technologies

Aided-view technologies consist of a special display and/or content plus a special user-worn stereo glasses. Using both the display/content and the glasses, the technology is able to direct the left and right views to the corresponding eyes. There are multiple types of these technologies, including the following.

Anaglyph: Anaglyph is the simplest case of aided-view technologies. Each lens in the anaglyph glasses has a different color such as red and cyan, and is used as a color filter. On the display side, while any display can be used in this technology, the content is manipulated in a way that matches...
the anaglyph glasses. That is, the red channel (R) of the left image is added to the green and blue channels (GB) of the right image to generate the output anaglyph image. The glasses filter out the red channel of the anaglyph image for the left eye and the other two channels for the right eye. The main advantage of the anaglyph technology is that it’s cheap and can work on any display. However, it suffers from a major drawback which is its degraded quality. Since some color information is lost during the content manipulation, the quality is degraded and 3D perception is reduced.

**Polarization Multiplexed:** Unlike anaglyph, in this technology a special display is required. The screen interleaves the left and right views using either row-interleaving or column-interleaving, and polarizes the light waves of each view in a specific direction. Passive glasses then filter the polarized light waves and allow only the light waves corresponding to each eye. Two types of polarized glasses are currently available: *linearly polarized* (one lens is horizontally polarized and the other is vertically polarized) and *circularly polarized* (one lens is polarized clockwise and the other counter-clockwise). Circular polarization allows more head tilt before the two views start mixing (cross-talk). The main advantage of this technology is that the glasses are cheap and light weighted. While the 3D quality is better than anaglyph, still half of the display resolution is lost due to interleaving. Polarized displays are most common in movie theatres.

**Time Multiplexed:** To improve the quality, active shutter glasses can be used. In this technology the display operates in double the usual frequency (usually 120 Hz) and temporally interleaves the two views by quickly alternating between the full resolution of both views. The glasses connect to the 3D display and filter the views by alternating the lenses from transparent to opaque in sync with the display. While no loss of information occurs in this technology, the main drawbacks are that the glasses are expensive and heavy, as they require battery to operate.

**Auto-stereoscopic Technologies**

Auto-stereoscopic technologies attempt to relieve the viewer from wearing glasses. In the autostereoscopic (glasses free) technologies, light rays are carefully directed to each eye. In order
Figure 2.3: In the autostereoscopic (glasses free) technologies, corresponding light waves are redirected to each eye by the use of (a) parallax barriers or (b) lenticular lens [10].

to perceive 3D the viewer should be positioned in a well-defined spot in front of the display unless eye-tracking devices are used. If the display is equipped with an eye-tracking device, it can prevent incorrect viewing by directing the right and left images to the appropriate spots. The two most well known auto-stereoscopic techniques are parallax barriers and lenticular lens.

**Parallax Barrier Displays:** Parallax barriers redirect the light by blocking it in certain directions. At the right distance and angle, each eye will only be able to see the corresponding view. The two main problems associated with parallax barrier systems are the loss of brightness, caused by the barriers themselves, and the loss of spatial resolution, caused by using only half of the pixels for each viewing zone. The optimum viewing distance in these systems is proportional to the distance between the display and the parallax barrier and is inversely proportional to the display pixel size. As the display resolution gets higher, the optimum viewing distance of the system gets longer [120]. Fig. 2.3 (a) illustrates the working principle of parallax barrier.

**Lenticular Lens Displays:** Lenticular lens redirect the light by reflecting it. An array of vertically oriented lenses are placed in front of columns of pixels. The alignment of the lenticular lens on the display panel is critical. This alignment gets more difficult as the display resolution increases and any misalignment can cause distortions in the displayed images [120]. The problem with lenticular systems is the reduction in resolution with the increase in the number of viewers. Fig. 2.3 (b) illustrates the working principle of lenticular lens.

Regardless of the technique used to direct the light, auto-stereoscopic displays can either be two-view displays, where only a single stereo pair is displayed, or multiview displays, where multiple stereo pairs are produced to provide different viewers positioned in different angles with different content based on their viewing angle. This enables multiple viewers to see the 3D objects from their own point of view, which makes these displays more suitable for applications such as computer games, home entertainment, and advertising. Content for these displays can be captured with multiple cameras or synthesized from two cameras.
2.2.3 3D Video Representations

A 3D video consists of multiple views and/or depth maps, and can be represented in different ways. In other words, there are multiple ways of merging the views/depth maps into a single video file. In the following we discuss the most popular representations.

Side-by-side

In the side-by-side representation, the left and right views are displayed beside each other. The two popular variations of this representation are top-bottom and left-right. Each of the left and right views have half spatial resolution, with the side-by-side being the same resolution as the 2D video. Therefore, it uses the same rate as the 2D.

Spatial-interleaving

In spatial multiplexing, the left and right views are interleaved spatially. The two variations of this representation are row-interleave and column-interleave. In row-interleave the odd and even rows each present a different view, while in column-interleave the odd and even columns present different views.

Temporal-interleaving

In this case, the left and right views are interleaved as alternating frames. Temporal interleaving has the advantage of maintaining the full resolution of each view. However, this comes at the expense of doubling the data rate.

Video Plus Depth

Unlike the previous representations that are tailored to a specific type of display with a fixed baseline distance between the two cameras, the video-plus-depth (V+D) representation provides more flexibility for customizing the 3D video at the receiver side.

The V+D format provides the 2D video along with the depth information of the scene in the form of an associated depth map video, as shown in Fig. 2.4. A depth map is a grayscale image representing the distance between the camera and each image pixel. Having the depth map enables the generation of virtual views, within a certain range around the captured view. This can be done using a high quality view synthesis technique such as depth image-based rendering (DIBR) [44].

The process of capturing the depth map is error-prone. One technique for capturing depth information is using the time-of-flight principle. Infrared laser beams are emitted to the scene, and the reflections are collected by the device to measure the time of flight [75]. This method works well for dynamic scenes, but it is yet to be determined how it will perform in some specific environments. For example, when there are mirrors, smooth/rough surfaces, very fast motion, heat/cold, etc. Depth
information may also be extracted from a stereo pair using stereo correspondences techniques. A survey on stereo correspondences techniques can be found in [105].

The V+D format has several advantages. First, encoding depth maps presents a small overhead on the video bit rate, which helps in minimizing the bandwidth. Second, it enables flexible adjustment of the 3D video to various display types and sizes, and different viewing preferences. However, this flexibility comes at the cost of increased complexity at both the sender and the receiver sides. Stereo correspondence algorithms, for example, are highly complex, time consuming, and error prone. Also, at the receiver side, it is required to synthesize a second view. Moreover, V+D is only capable of rendering a limited range of views and is prone to errors at disoccluded points.

**Multiview Plus Depth**

One key issue with V+D is that it enables synthesizing only within a limited range around the original view. This is due to the *disocclusion* problem, where some regions in the virtual view have no mapping because they were invisible in the original reference view. As a result, the virtual view will have some holes and requires applying a filling algorithm that interpolates the value of the unmapped pixels from surrounding areas. This disocclusion effect increases as the angular distance between the original view and the virtual view increases. In order to have a large number of output views at the decoder side, the multiview-plus-depth (MVD) format was introduced. This format contains multiple 2D views along with their associated depth maps.

Virtual views may be synthesized more correctly if two or more reference views, from both sides of the virtual view, are used [54]. This is possible because areas which are occluded in one of the reference views may not be occluded in the other one. Fig. 2.5 shows an example of this process using only two reference views.
2.2.4 Principles for Producing Good-quality 3D Content

In order to create an accurate, comfortable and compelling 3D effect (whether using 3D cameras or 2D-to-3D conversion) many factors should be considered. Here we will discuss some of the important rules and principles for proper 3D content creation. While most of these principles are primarily used for cinematography, non-cinema 3D content can also benefit from them [83].

1. In order for the stereoscopic content to be comfortable to watch, every object in the scene needs to fit within the comfort zone, i.e., it cannot be too far from the screen plane. Comfort zone mainly depends on the screen size and viewing distance. A rule of thumb used by stereographers (known as the percentage rule) is that the pixel disparity should not exceed 2–3 percent of the screen width [107].

2. The relative depth between nearby objects cannot be too large, i.e., the disparity needs to be within the limit known as Panum’s fusional area [25]; otherwise, the observer will not be able to fuse one of the objects.

3. In order for the two views to fuse properly, both views should be identical in all characteristics except for the slight disparity between them. Any other difference such as color, lighting, timing, focus, or image geometry can overload the visual system, causing less impressive 3D or discomfort such as eye strain and headaches [83].

4. The transition in depth from one shot to another should avoid sharp changes in depth. For example, if one shot is behind the screen, and the next shot is far in front of the screen, the viewer’s eyes must quickly re-converge and re-focus, which will most likely cause discomfort.
As a result, it is best if shot transitions move fluidly from one point of convergence and depth to the next [118].

5. It is important to avoid retinal rivalry. Retinal rivalry occurs when one eye sees part or all of an object and the other eye does not. This effect usually causes the object to appear slightly transparent because it exists in one eye and not in the other [118].

6. Retinal rivalry can also cause stereoscopic window violation. Window violation occurs when the object that is seen by only one eye is interpreted as being in front of the screen. Since if the object is in front of the screen, it should not be occluded by the edge. In 3D cinematography this problem is often solved by floating the stereoscopic window which applies masks on the sides of the frame to hide what the eyes should not see [83].

7. A well choreographed depth should also support the emotional journey of the story. This is usually referred to as the Depth Script. For example, it can be compelling if the depth range remains small in less emotional scenes and grows for more emotional scenes [118].

8. In order to give the sense of volume to objects rather than just flat cards in space, depth should change along their shape [118].

9. Since in the real world the apparent volume reduces as the object moves away from the viewer, this affect should be recreated in an accurate 2D-to-3D conversion [118].

2.3 VR Content Representations

VR videos are captured using 360 camera rigs, which capture the scene from different angles. The rigs contain multiple cameras placed next to each other such that their collective field of view would cover the 360 space while having enough overlap between them for later stitching purposes. To generate the 360 video, video streams from all the cameras are synchronized, stitched, and projected on a sphere. Multiple spherical projections have been proposed, including equirectangular projection, cubemap projection, pyramid projection, and offset-cubemap projection.

2.3.1 Equirectangular Projection

The equirectangular projection is a standard way of projecting the 3D world onto a flattened sphere. It is an image with size $2\pi r \times \pi r$ that will be wrapped around a sphere when viewed in 360 degree. Note that $r$ is constant for all points, and can be chosen arbitrary based on the desired output size (resolution). Fig. 2.6 demonstrates the equirectangular projection. The most known example for equirectangular projection is the world map. The equirectangular projection is widely supported and easily viewable even with no special players. However, its main drawback is that it contains redundant information. The equirectangular projection stretches the image as we move towards the poles, causing a lot of redundant pixels that will waste the user’s limited bandwidth in a streaming scenario.
2.3.2 Cubemap Projection

In order to address the problem of redundant information, the cube map projection was proposed. Unlike equirectangular projection, cubemap doesn’t have stretched areas. An open source implementation for an ffmpeg [46] filter that converts a video mapped as an equirectangular to cubemap was released by Facebook [17]. For a cubemap projection a sphere is first projected on a cube and then flattened by placing the six sides of the cube next to each other. Cubemap projection can be easily rendered using graphics libraries and has been used extensively in gaming applications.

2.3.3 Pyramid Projection

The main idea of pyramid projection is to project the sphere on a pyramid where its base is the user’s current viewing area. As a result, the user’s viewport will have the highest number of pixels. It also allows for a smooth degradation of quality as the users move their head to the back. However, one of the main drawbacks of this projection is that it is not supported on GPUs, and thus its rendering is not as efficient as cubemap.

2.3.4 Offset-cubemap Projection

Offset-cubemap projection builds on top of the cubemap to provide smooth degradation of quality while addressing the issues of the pyramid projection. Offset-cubemap is a regular cubemap where the user is pushed back from the center of the cube with a predefined offset. Since the offset-cubemap is essentially a cubemap, it can be efficiently rendered on a GPU. In addition, offset-cubemaps offer a smoother degradation of quality compared to pyramid projection. Facebook recently released the implementation of the offset-cubemap as part of the cubemap open source code [17], however, it is not yet widely adopted.
2.4 3D Video Streaming

Multimedia streaming is typically referred to the case when video is decoded and displayed at the client, while being received from the servers. In order to make efficient use of the bandwidth, compression techniques are used prior to the transmission. Using higher compression ratios while encoding can reduce the bitrate, but it can cause degradation in quality. When streaming a video from server to client over packet delivery networks many parameters should be taken into account such as throughput, loss and delay. For example, if the video bitrate increases the network throughput, quality degradations can be caused due to loss, while a low bitrate can also degrade the quality due to compression. Managing this trade-off in order to maximize the quality at the receiver side is usually one of the main issues in multimedia streaming, which rate adaptation techniques attempt to solve. In 3D and multi-view media streaming becomes even more complicated as it requires larger bandwidth, and more dependencies can be leveraged while encoding it. In addition, since 3D and multi-view media include multiple components such as different views and/or associated depth maps, new methods can be introduced for their encoding, such as compressing the depth maps more than the video, and/or compressing some of the views more than others. For example, if the two views do not have the same spatial/temporal resolution, the human visual system perceives high frequencies in 3D from the view which has a higher quality. We can make use of this fact by subsampling one of the views on the server-side, either spatially or temporally to reduce the overall transmission rate. At the receiver side, the subsampled view will be interpolated to full resolution once again before displaying. Therefore, existing 2D video streaming schemes cannot be applied directly for streaming 3D videos, and efficient and intelligent solutions are needed. Rate adaptation can also be performed with feedback from the client. For example, one possible scenario would be for the client to track the viewer’s head position and select only a small number of views according to the position.

In addition, there has been a significant interest from the industry in 3D video streaming, such as YouTube, 3DVisionLive, Trivido, and 3DeeCentral. YouTube [15] supports multiple 3D formats including anaglyph (red-cyan, blue-yellow or green-magenta), side-by-side, row and column interleaved. It also supports HTML5 stereo view, which is the format for active-shutter displays that utilize NVIDIA 3D Vision. 3D content on autostereoscopic mobile devices such as LG Optimus 3D cellphone is also supported by YouTube. 3DVisionLive [2] is a web channel for 3D video streaming and 3D photo sharing. 3DVisionLive uses the Microsoft Silverlight and IIS smooth streaming technologies. It is, however, limited to one display technology. In order to view 3D content on 3DVisionLive, NVIDIA active-shutter glasses are required. Trivido [14] is a 3D Internet video platform. It supports anaglyph, side-by-side, and row-interleaved 3D formats in addition to 3D NVIDIA Vision format. Trivido supports only two display technologies: polarized and active-shutter, and it does not support autostereoscopic displays. 3DeeCentral [1] supports 3D content on multiple 3D-enabled devices. Five different classes of displays are supported in the system: Internet-Connected 3D TVs, Windows 7 desktop devices, Windows 7 laptops, Android auto-stereoscopic devices such
as LG Optimus, and HTC Evo 3D, iOS devices such as iPhone 4, and iPhone 4S. Despite the support of different types of devices, it is not clear whether 3DeeCentral provides adjusted depth for different sizes.

### 2.5 3D Video Customization

Depth customisation is an important factor in having a compelling 3D experience. Without a proper customization the 3D quality can degrade significantly even if transmission has been performed perfectly. The problem is that most 3D content was originally designed to be viewed on one type of display, in cinemas. As a result, if such content is streamed to other displays it might not be compelling or comfortable any more. As discussed in Section 2.2.4, in order for a 3D video to be comfortable its depth should stay within the comfort zone. Since the comfort zone is a function of the viewing condition (i.e., display size and viewing distance), a video designed for a larger comfort zone may not be comfortable for smaller ones. In addition, a video designed for a small comfort range will look rather flat on a display with a larger comfort zone. This issue becomes even more important given that currently the 3D displays available in the market have various sizes such as, mobiles, tablets, laptops, desktops, and TV sets.

In [107], the effect of viewing distance and direction of the vergence-accommodation conflict on discomfort is studied for a wide variety of situations including the viewing of mobiles, desktops, TV, and cinema. Vergence-accommodation conflict is one of the common reasons of discomfort when viewing on stereo displays. It is created because the focal distance of the eyes is fixed at the distance from the display, while vergence distance depends on the distance being simulated on the display. Results show that while content behind the screen is less comfortable at far distances, content in front of screen is less comfortable at near distances.

In addition to viewing conditions, the depth perception varies from person to person [38, 40]. While some people might enjoy an exaggerated depth, others can experience discomfort with even small amounts of depth. As a result, depth should be customized for each user individually.

Meeting the comfort zone constraints is dependent on two stereoscopic parameters: the camera separation and the convergence. Camera separation influences the range of depth (and thus distances between objects), while convergence influences the placement of that range relative to the screen plane. With modifying these two parameters the content can be customized to any viewing condition. However, these parameters can only be modified during the content creation. Oscam et al. [90] developed a method that optimizes these parameters in real-time. However, it is limited to synthetic content, and cannot be applied as a post-process.

The convergence is relatively easy to modify in post-production. This can be done by simply shifting all the image pixels a few pixels to the left or right. It can be performed on either one or both views. Depending on the direction of this shift the whole scene will move towards the screen or further away from it.
Modifying the camera separation, however, is rather difficult, since it requires synthesizing new camera views. In order to synthesize a new view, depth map is required. With the depth map provided, the new views can be synthesized to be in any desired depth range. Lang et al. [77] show how this can be accomplished via nonlinear disparity mapping. However, for generating the depth map, stereo correspondence techniques are required, which are usually time-consuming and prone to various artifacts. Therefore, it is difficult to reliably generate clean, artifact-free output without manual intervention.

In addition to depth customization, 3D videos should be customized based on the type and technology of the display as different displays require different input formats. Different 3D formats have been discussed in Section 2.2.3.

2.6 2D-to-3D Conversion

While customization methods adapt the video to different viewing conditions and personal preferences, they require 3D videos as input. However, capturing 3D content is far more costly than regular 2D videos. It requires expensive 3D cameras, staff training and involves higher transportation and setup costs. This becomes even more challenging for live events, such as soccer games, that require multiple stereo rigs in the stadium to cover all aspects and angles. Therefore, many production companies find it challenging and thus rarely attempt to upgrade to 3D. As a result, the lack of high-quality 3D content becomes another challenge that hinders the wide spread adoption of 3D videos. In addition to new content, there is a high demand in viewing many of the already existing 2D videos in 3D. As a result, there has been a huge interest both in academia and industry in 2D-to-3D conversion methods.

Since 3D perception requires two views, 2D-to-3D conversion methods generally estimate the second view based on the first view (the original 2D video). In order to do so, the depth map should first be estimated. If the depth map is available the second view can be rendered using DIBR techniques as described in Section 2.2.3. As a result, the main challenge for 2D-to-3D conversion techniques is depth map estimation, and different conversion methods mainly differ in their depth estimation technique.

In general, 2D-to-3D conversion methods can be divided into three categories: manual, semi-automatic, and automatic methods. Most movies are up-converted to 3D manually, meaning that depth maps are painted manually by stereographers and used to render the corresponding views. This process yields very good results but it is extremely costly and time-consuming. In semi-automatic methods human intervention is still required but largely reduced. Human intervention may be in the form of: (i) requiring initial inputs from the user such as user-defined scribbles corresponding to a rough estimate of the depth values [123, 57, 101], (ii) producing an initial depth such as the background depth and requiring the user intervention for foreground depth assignment [103], or (iii) allowing the user to interactively refine the depth and object boundaries through the process [136]. Automatic methods can operate without the use of any intervention. They are more
Current depth estimation techniques can be divided into two categories: Depth cue based techniques, and data-driven methods. In the following we will first discuss different depth cues, followed by a discussion on data-driven methods.

2.6.1 Depth Cues

In order to retrieve depth information from a 2D content, depth cues can be used. As humans, there are a variety of depth cues present in each 2D scene that help us realise the depth distribution in that scene. Many of these depth cues can also be used to estimate the depth automatically. Some of the common cues are motion parallax, focus/defocus, light and shading, linear perspective, and height in image.

Motion parallax techniques such as [103] are based on the fact that in a static scene (e.g., Fig. 2.7(a)) with a freely moving camera, near objects move faster than far objects. Thus, the main
drawbacks of this approach are: (i) if the camera has no motion, there will be no motion parallax, and (ii) if there are independently moving objects in the scene, errors can occur.

Focus/defocus techniques such as [121, 55] are based on the assumption that the foreground objects are in focus and the background ones are blurred. Depth is then assigned based on the levels of blur. Fig. 2.7(b) shows an example image that can benefit from defocus. In this image the man is in focus, while the background trees are blurred. Also, the further trees are more blurred than the near ones. Using defocus on this image will result an accurate depth map with the man in front and the trees far away from the display. The main drawback of this approach is that the foreground and background will not be distinguishable if the amount of blur is similar.

In the light and shading approach, the information provided by shadows is utilized to extract the light source and recover the shape of the reflecting surface. Fig. 2.7(a) shows an example image that can benefit from shading. Based on the shadows in this image, the light source is at the bottom-right side of the scene. The shape and structure of the heads can also be detected based on their illumination. A good survey for these techniques can be found in [134].

In linear perspective parallel lines converge or appear closer to each other as they move to the distance and get deeper. If the linear perspective cue is present in a scene, depth can be estimated by detecting the parallel lines in the image, identifying the vanishing point (the point where these lines converge), and assigning depth based on the position of the lines and the vanishing point [64]. Fig. 2.7(c) shows an example image that can benefit from perspective. In this image the three parallel lines on the bottom-right, bottom-left, and top-left get closer as they reach the end of the hall. The door is the vanishing point and the deepest part of the scene.

Height in image is based on the assumption that objects closer to the bottom of the image are closer to the camera, and the distance will increase as we move towards the top of the image. This is usually the case in outdoors and landscape scenes but cannot be applicable in most of the indoor scenes [70, 36, 106].

In order to have a more robust depth estimation, several cues can be combined together. For example in [64], motion parallax and linear perspective are used together. A depth map is first estimated using each of the cues individually, and then fused together. In [136] a semi-automatic 2D-to-3D conversion system is proposed based on a combination of depth from motion, depth from defocus, and depth from aerial perspective. Similar to [64], three depth maps are first estimated using each individual cue, and then linearly fused together to compensate the weaknesses of each other.

### 2.6.2 Data-driven Methods

In addition to the traditional methods discussed above, more recently data-driven methods were introduced, which provide an alternative way of estimating depth maps and the corresponding stereo views. Data-driven methods are based on 3D data rather than depth cues. Instead of identifying the appropriate depth cues, depth is inferred from a 3D database. However, the database should be
related to the content and must include similar data. In addition, generating a 3D database is one of the main challenging issues in this approach.

Konrad et al. [74] use Youtube 3D images as their database. Having a huge database of image and depth map pairs, they infer depth for an input image. Their work is based on the assumption that images with similar gradient-based features, e.g., HOG (Histogram of Oriented Gradients), tend to have a similar depth. Therefore, for a query image, the most similar images from the database are found based on the chosen gradient-based feature. The query image depth is then estimated as the median over depths of the retrieved images. Fig. 2.8 shows the block diagram of this algorithm.

Karsch et al. [71] extended this approach to videos. They generate a large database of image and depth map pairs using Kinect data. Similar to [74] for a query frame, they first find the most similar images in the database. They then warp the retrieved images to the query image using the SIFT flow algorithm [82]. This warping is applied to the depth maps as well. Finally, the warped depth maps are combined to estimate the final depth, using an optimisation function. Fig. 2.9 shows the block diagram of this algorithm.

While current results of the data-driven approach are promising, still many challenge remain unsolved, such as generating a high-quality dataset for outdoor scenes where Kinect fails. In addition,
Figure 2.9: Block diagram of the overall algorithm for 2D-to-3D conversion in [71].

current techniques are unable to handle scenes with complex motion, and different scenes from the data available in the database.

In addition to the academic work, some commercial products have also explored automated 2D-to-3D conversion. Some are sold as stand-alone boxes (e.g., JVC’s IF-2D3D1 Stereoscopic Image Processor, 3D Bee), while others are software packages (e.g., DDD’s TriDef 3D). While the details of these systems are not known, their depth quality is still an outstanding issue [133].
Chapter 3

Customization and Streaming of Stereoscopic Sports Videos

3.1 Introduction

Compared to standard 2D videos, stereoscopic 3D videos provide a more entertaining experience [49]. Due to the significant interest in such content, most of the recent big movie productions are either shot in 3D or converted to their stereoscopic versions in post-production. At the same time 3D displays have become a commodity. Most of the off-the-shelf TV-sets are 3D-ready. Also, several laptops and mobile devices are equipped with stereoscopic displays [108]. This trend of incorporating stereoscopic technology into home entertainment systems may be easily hampered by the challenges in streaming 3D videos. The main problem is that 3D streaming systems must be capable of serving stereoscopic content to a wide range of display devices that are used in uncontrolled conditions. As both depth perception and visual comfort highly depend on viewing conditions (e.g., display size and viewing distance) [115, 117, 107], streaming one version of the content (e.g., a movie theater copy) is sub-optimal. In addition, users have different preferences in terms of how much of the stereoscopic effect should be present in the content. This suggests that additional methods that enable easy content personalization are a necessary feature of any streaming system.

To address these issues, we propose a system for stereoscopic 3D video streaming. In contrast to previous systems, it allows users to personalize 3D content based on their own preferences, which significantly improves their viewing experience. In particular, we propose a novel method for depth expansion and compression of stereoscopic 3D videos. Our method performs simple image processing operations, it does not require creating accurate depth maps, and it does not introduce visual artifacts. The main target application for our method is sports videos, e.g., soccer, football, tennis, and baseball. In such applications, preserving the scene structure, e.g., the straightness of the field plane and the white lines, is extremely important. Our method guarantees to preserve the scene structure, because we employ linear operations to map the original image coordinates to new ones. In addition, we propose a method for depth personalization, which allows users to control the
depth in 3D videos to suit their comfort level. This is an important feature as not all users prefer the same amount of depth, some users do not like depth at all, while others are fascinated by 3D content with striking and aggressive depths.

In this chapter, first in we summarize previous efforts in academia and industry in designing 3D video streaming systems, and also describe different 3D content customization methods and how our method is different from them in Section 3.2. Then, we present the design of our system (Sec. 3.3) and the depth customization method (Sec. 3.4), followed by our user studies which evaluate our system in Section 3.5. Finally, Section 3.6 summerizes the chapter.

3.2 Related Work

There has been significant interest both in academia and industry in 3D video processing. However, as discussed below, the problem of depth customization for different display sizes and types has not received much attention. In addition, in contrast to previous methods, our depth manipulation method does not rely on accurate depth maps, it does not introduce any significant distortion to the videos, and it is computationally inexpensive.

3.2.1 3D Streaming Systems

Multiple systems for 3D content streaming have been proposed in the literature. The ATTEST project [102] aims to provide the full pipeline of 3D-TV broadcasting (e.g., content generation, coding and transmission, displays, and perceptual evaluation) while being backward compatible with 2D. ATTEST considered certain input format (i.e., V+D) captured by special devices, and it employed DIBR methods for depth customization [43]. Xin et al. [127] describe a 3D video streaming system, but their main focus is on the 3D media encoder and decoder. Carballeira et al. [31] also focus on encoding and propose a framework to analyse and reduce the encoding latency of multi-view videos. Gurler et al. [56] discuss 3D formats and coding for different streaming architectures, and consider rate adaptation for P2P multi-view streaming and selective-view streaming. Vetro et al. [122] focus on compression and representation of multi-view video. Diab et al. [39] focus on optimizing the storage in 3D streaming systems. Pehlivan et al. [95] propose a video streaming system that switches between 2D and 3D videos depending on the available bandwidth and display equipment. The work by Wu et al. [126] adapts 3D video quality and balances/trades off the temporal and spatial (color plus depth) quality in real-time, but it does not enhance or customize the depth signal for different display types/sizes/technologies. Our work could potentially leverage their method to provide smarter rate adaptation in our 3D streaming system. 3D teleconferencing has been also proposed. For example, Johanson [69] focuses on extending the transport protocol to associate left and right views. Whereas the more recent ViewCast [129] enables multi-party 3D tele-immersion and prioritizes stream transmissions based on the client’s viewing angle. Depth customization is not addressed in these works. In addition, multiview client-server systems, where a scene can be displayed from different viewpoints has been considered, for example in [72] and [59].
In addition to the academic works mentioned above, there has been significant interest from the industry, including YouTube, 3DVisionLive, Trivido, and 3DeeCentral. YouTube [15] supports multiple 3D formats including side by side, anaglyph (red-cyan, blue-yellow or green-magenta), row and column interleaved. However, unlike our system, it does not customize or change the video depth for different displays. 3DVisionLive [2] is a web channel for 3D photo sharing and 3D video streaming, but it is limited to only one display technology. Trivido [14] is a 3D Internet video platform, which supports side by side, anaglyph, row interleaved, and the 3D NVIDIA Vision format. However, the problem of content customization for different displays has not been addressed in this platform. 3DeeCentral [1] supports 3D content on multiple 3D-enabled devices. However, depth adjustment for different displays is not provided.

In summary, previous works on 3D streaming have addressed various problems including content representation, content acquisition, encoding, and transmission of both stereoscopic and multiview content. However, we are not aware of any 3D streaming system that adaptively customizes the depth based on the display technology, display size, and viewer’s preferences.

3.2.2 Depth Customization

As described in Sec. 2.2.4, there are two basic requirements that stereoscopic content should meet in order to be comfortable to watch. First, every object in the scene should fit within the “comfort zone”, i.e., it cannot be too far from the screen plane [107]. A rule of thumb used by stereographers (known as the percentage rule) suggests that the pixel disparity should not be greater than 2–3 percent of the screen width [107]. Second, nearby objects (in xy-plane) cannot be too distanced from each other in the z-direction, i.e., the disparity should be within the limit known as Panum’s fusional area [25]; otherwise, the observer will not be able to fuse one of the objects. Meeting these constraints is dependent on two stereoscopic parameters: the camera separation and the convergence. Camera separation influences the range of depth (and thus distances between objects), while convergence influences the placement of that range relative to the screen plane. Oskam et al. [90] developed a method that optimizes these parameters in real-time. However, it is limited to synthetic content, and cannot be applied as a post-process.

Modifying the convergence in post-production is relatively easy and can be accomplished by simply shifting either one or both views. Fixing the camera separation, however, is more difficult, since it requires synthesizing new camera views. Lang et al. [77] show how this, and even more sophisticated operations, can be accomplished via nonlinear disparity mapping. Unfortunately, their method relies to a large extent on stereo correspondences, and it requires recovering pixel values for points that have not been registered by the camera. Therefore, it is difficult to reliably generate clean, artifact-free output without manual intervention, not to mention real-time performance. Furthermore, nonlinear disparity mapping can severely degrade the quality of videos of field sports such as soccer, due to objectionable curving of the lines. Depth can also be manipulated as a consequence of depth compression performed to limit the bandwidth of 3D content. Although some of these techniques can adapt to different viewing conditions [126, 92], their primary goal is to main-
tain the original depth. In contrast, our technique intentionally modifies the depth to enhance viewer experience.

3.3 System Architecture

The proposed 3D streaming system, provides depth-optimized videos to clients with different 3D display types and sizes. As depicted in Fig. 3.1, the proposed 3D streaming system consists of: (i) server that processes the stereoscopic content and creates an optimized version for each display based on its size and technology, and (ii) multiple clients from which 3D devices request display-optimized versions and receive the corresponding ones.

**Server Side:** The server adjusts the depth of 3D videos and adaptively streams them to clients upon their requests. It also creates and stores multiple versions of the 3D videos to cater different display requirements. In particular, the server has four main components: (i) Depth Enhancer, (ii) 3D Version Manager, (iii) Storage Manager, and (iv) Adaptive Streamer.

The Depth Enhancer customizes the depth of original 3D videos based on the target displays. It can either increase or decrease the amount of perceived depth in the video by employing lightweight image processing methods. We will describe this component in more details in Sec. 3.4.

The 3D Version Manager creates different versions of the same 3D video. Specifically, the input video is assumed to be in the side-by-side format, which is currently the most commonly used 3D representation. The proposed depth enhancement method creates up to \( D \) versions with different depth values, where \( D \) is the number of depth levels supported by the system. All \( D \) versions are still in the side-by-side format. Next, various video conversion methods are applied in order to support displays with different depth rendering technologies, which include anaglyph, frame interleaved, row interleaved, column interleaved, and video plus depth. For each of the \( D \) versions, up to \( R \) versions are created, where \( R \) is the number of different 3D video representations

---

1 This system was developed in collaboration with Khaled Diab and Tarek Elgamal.
supported by the system. In addition, we can create up to \( B \) versions for each previous version to accommodate clients with heterogeneous and varying network bandwidth, where \( B \) is the number of different bitrates (qualities) offered by the system.

The Storage Manager manages the storage of different versions. The proposed 3D Version Manager can create up to \( D \times R \times B \) different 3D versions. With current display technologies and sizes, the parameters \( D, R, \) and \( B \) are roughly in the ranges \([5 – 20]\), \([5 – 7]\), and \([10 – 20]\). That is, more than 200 versions of the same 3D video are required to support the large diversity of clients. Creating and storing all versions for all videos may waste storage and processing resources of the system, especially for videos with low popularity.

Adaptive Streamer, the last component of our system, adopts the Dynamic and Adaptive Streaming over HTTP (DASH) protocol [66, 110, 113]. DASH enables the server to scale to many concurrent sessions, uses commodity HTTP servers, facilitates switching among different versions, and provides wide client accessibility because it uses the standard HTTP protocol which is allowed in most networks. The server side of the DASH protocol divides a video into a sequence of small video segments. Current 2D streaming systems that use DASH create few versions of each segment at different bitrates. In our 3D streaming system, however, we create different versions of each segment using the 3D version tree to adapt not only to the network bandwidth, but also to different display sizes and technologies. Using the proposed depth expansion/compression method (Sec. 3.4) and the adaptive nature of DASH streaming, we propose the feature of \textit{depth personalization}, which allows adjusting the depth of a video based on the preferences of individual viewers. This is a desirable feature, as it improves the engagement of viewers and allows them to choose the most visually comfortable depth perception.

\textbf{Client Side:} As shown in Fig. 3.1, our system supports multiple client platforms, including: (i) mobile, (ii) web-based, and (iii) custom. These clients implement the client side of the DASH protocol to adaptively request segments of the different versions of 3D videos stored at the streaming server. Specifically, the client starts by requesting the manifest file from the server. The manifest file contains meta data about the requested 3D video and its available versions. Then, based on the display characteristics and the current network conditions, the client decides on the most suitable 3D version. The client then starts requesting video segments from the server and the media player is notified to start playing these segments. For the mobile client, we developed an application that works on various mobile devices such as smartphones and tablets. Our web-based client employs standard web interfaces to render the encoded 3D video, while the 3D display along with its associated glasses (if any) creates the 3D perception for the viewer. Finally, there are displays that require special handling of 3D videos, such as the multi-view displays. Such displays usually require the video in the form of video-plus-depth (2D images and their associated depth map). View synthesis methods are then used to create multiple virtual views from the video-plus-depth input. In our system, we implemented a custom client for such displays, which asks for video-plus-depth segments from the server and performs view synthesis using the manufacturer APIs.
3.4 Depth Customization and Personalization

We need to show a different view to each eye to perceive binocular stereopsis. These two views show the same scene but from two slightly different viewing angles. Specifically, for each pixel in the left view, its corresponding pixel in the right view is located a few pixels away. Given a pair of corresponding pixels, the signed distance \( d = x_r - x_l \) between their x-positions is called disparity. Disparities are detected by the human visual system and interpreted as depth. The depth perception from the display plane depends on the disparity value. An object is perceived behind or in front of the display plane if the disparity is positive or negative respectively. The object appears on the display plane in the special case of zero disparity. The amount of perceived depth is a function of disparity, viewing distance, and inter-ocular distance. Larger disparities result in perception of larger distances [63].

In addition to viewing conditions, the depth perception varies from person to person [38, 40]. Thus, there is a need for techniques enabling customization of the depth distribution in 3D videos. In the context of streaming sports videos, such techniques need to meet three requirements: (i) they need to work as a post-process, since we do not have influence on the recording process, (ii) they need to be fast (in case of live streaming), and (iii) they need to automatically produce high quality results, without objectionable artifacts. In the following subsection, we propose a new method for depth expansion/compression that meets these requirements. It targets videos of various field sports such as soccer, football, and tennis. In Sec. 3.4.2, we utilize this method to enable depth personalization.

3.4.1 Structure Preserving Scene Shifting

Depth adjusting for stereo videos is a non-trivial task. Unlike video-plus-depth content, which provide more flexibility for depth customization during view synthesis, customizing a stereo video requires disparity remapping. This is, however, a viable option only in off-line applications with some form of supervision. The only safe automatic adjustment for general scenes is convergence manipulation, which can be easily performed using a horizontal shift of the two views. We observed, however, that for some scenes, especially in sports videos, the geometry has approximately planar structure. In such cases, depth maps can be well described by a single depth gradient \( g = (g_x, g_y) \).

Based on the previous observation and discussion, we propose the Structure Preserving Scene Shifting (SPSS) method for depth expansion/compression, which adjusts the depth range of the scene by means of 2D geometric transformations. The basic idea behind the SPSS method is to estimate the depth gradient \( g \) and adjust its magnitude. The gradient is estimated by fitting a plane to the scene’s depth map which is obtained using a stereo-correspondence algorithm. In contrast to the disparity remapping approach, we use stereo correspondences only to estimate the overall depth gradient, hence the accuracy of the correspondences is not critical.

Modification of the gradient is achieved via a remapping operation, in which a parallelogram-shaped region of the input image is selected and mapped back onto the whole image area. Such a
Figure 3.2: Example of the slant operation. A parallelogram-shaped region is mapped back onto the whole image area, modifying the vertical component \( g_y \) of the depth gradient \( g \).

mapping can be applied to one of the views or both, and in the latter case, the depth modifications caused by each transformation will add up. To minimize visibility of the distortion, we split the desired depth adjustment between the two views, so that each mapping of one view is accompanied by a complementary mapping of the other. The top and bottom edges of the mapping parallelogram are always kept horizontal, and its shape is described by a combination of two parameters: the slant and the stretch.

The slant operation regulates the skewness of the mapping region by horizontally moving its top and bottom edges in opposite directions. This operation modifies \( g_y \). An example is given in Fig. 3.2. The stretch operation re-scales the mapping region horizontally. This transformation modifies \( g_x \). An example is given in Fig. 3.3.

The slant and stretch operations are done linearly to assure that any plane structure remains planar. This becomes especially important in sports videos since deformations such as curvatures in the field plane or the white lines can be very visible and disturbing.

We denote the amount of slant and stretch as \( \sigma_{sl} \) and \( \sigma_{st} \), respectively. Assuming that the image x- and y-coordinates are in the range \([-\frac{1}{2}, \frac{1}{2}]\), the two operations are combined into one operator, mapping a pair of old coordinates \((x, y)\) to a pair of new coordinates \((\hat{x}, \hat{y})\) as shown in Eq. (3.1). To accommodate slanting and stretching, the mapping region has to be slightly smaller than the input image, therefore the factor \( r \) (typically 0.95) is used to re-scale the coordinates. Recall, that the depth transformation is split equally between the two views of the stereo image, hence the factor \( \pm 0.5 \). In addition to slanting and stretching, a shift operation that moves the mapping region to the
Figure 3.3: Example of the stretch operation. The left and right views are horizontally scaled. In effect, the horizontal component $g_x$ of the depth gradient $g$ is changed.

left or right can be used to adjust the convergence. Depending on the direction of the shift, the scene disparities are uniformly increased or decreased, and as a result, the scene appears to pop-out of the display (i.e., negative disparity) or go deep behind the display plane (i.e., positive disparity). We refer to the amount of shift as the \textit{pop-out factor}, and we denote it by $\beta$. The default value for $\beta$ is zero. In Sec. 3.4.2 we discuss the effect of assigning different values to $\beta$.

\begin{align}
(\hat{x}, \hat{y}) &= (x \pm 0.5 \cdot (\sigma_{sl} \cdot y + \sigma_{st} \cdot x + \beta), y) \cdot \frac{1}{r}, \quad (3.1)
\end{align}

From the equation above, we can infer that the maximum added disparity is:

\begin{align}
\max_{-\frac{1}{2} \leq x \leq \frac{1}{2}, -\frac{1}{2} \leq y \leq \frac{1}{2}} \sigma_{sl} \cdot y + \sigma_{st} \cdot x = \frac{|\sigma_{sl}| + |\sigma_{st}|}{2}. \quad (3.2)
\end{align}

This value is the maximum added disparity in the positive direction and the negative of this value is the maximum added disparity in the negative direction.

\textbf{SPSS Coverage:} Ekin et al. [41] distinguish four types of camera shots in soccer games: long shots, medium shots, close-ups, and out-of-field shots. Long shots provide a global view, in which the field takes most of the display-space, and multiple small player silhouettes are visible. Medium shots show a smaller portion of the field, usually with couple larger silhouettes, while close-ups zoom on one or few players only. Finally, out-of-field shots show the audience, coaches, etc.
Figure 3.4: Examples of shots in a soccer video that have planar depth structure. The depth maps were determined using optical flow estimation methods [119, 24], and were further enhanced by cross bilateral filtering [93]. Note, that they are provided for visualization purposes, and our method does not require computationally expensive estimation of accurate, dense stereo correspondences.

Figure 3.5: Our method has increased the disparity range while preserving the depth structure and orientation (the top images are the original disparity maps, while the bottom ones show depth after our optimization.)
Close-ups, out-of-field, and medium shots usually have quite complicated geometry. Long shots, however, are different, because their geometry can be very well approximated by a plane, therefore, they are perfect candidates for the SPSS (see Fig. 3.4a for an example of a long shot). Occasionally, some medium or out-of-field shots, such as the one shown in Fig. 3.4b, are also well fitted by a plane, and thus can benefit from the proposed depth adjustment. In the evaluation section, we analyze the shot types in different sports videos and show that the proposed SPSS method can cover a significant portion (60–70%) of the shots in field sports.

Our depth adjustment technique assumes that the scene can be well approximated by a plane. As a result, to detect scenes suitable for our technique, we should estimate how well a scene can be approximated by a single plane. To this end, we first recover depth using \[128\]. Next, we fit a plane to it using the least squares method and compute the coefficient of determination (\(R^2\)) to measure its goodness. We then construct a binary classifier, which classifies a scene based on the \(R^2\) value, i.e., if the goodness is above a certain threshold \(q\), the scene is considered suitable for our manipulations. Otherwise, it remains unchanged. In order to determine a good threshold \(q\), we processed 1,015 randomly chosen frames from one soccer game and classified them manually. Then, we analyzed how our binary classifier performs for different threshold values using the relation between true positive and false positive ratios. Fig. 3.6 presents the ROC curve. Based on the analysis we chose \(q = 0.693\), which gives true positive ratio = 0.8981, and false positive ratio = 0.1925.

**Main Steps of SPSS:** For any scene classified as suitable for the SPSS method, the following steps are taken:

1. Compute the gradient \(g = (g_x, g_y)\) of the scene’s disparity map.
2. Compute the slant and stretch as follows, where $\hat{d}$ is the target disparity, which depends on the viewing conditions and the content.

$$
\sigma_{sl} = \hat{d} \cdot \frac{g_y}{g_x + |g_y|} - g_y, \quad \sigma_{st} = \hat{d} \cdot \frac{g_x}{g_x + |g_y|} - g_x, \tag{3.3}
$$

3. Temporally smooth $\sigma_{sl}$ and $\sigma_{st}$ using the history of their values in the $n$ previous frames in order to maintain temporal coherency.

4. Remap the views according to Eq. (3.1) using linear interpolation.

Fig. 3.5 shows the depth maps of the two samples from Fig. 3.4, before and after SPSS. It can be seen that SPSS enhances the depth contrast, while preserving the original scene structure. In both samples, the depth difference between points A and B has increased, while the direction of the depth gradient has remained unchanged.

**Remarks and limitations of SPSS:** We note that SPSS preserves the straight lines in the scene and does not introduce artifacts, because Eq. (3.1) is a linear remapping of input coordinates. Although linear remapping operations such as scaling and stretching can cause quality degradation due to pixel averaging, in SPSS this degradation is negligible. This is because the target disparity and thus the amount of slanting, stretching, and scaling performed in SPSS is small. Increasing the disparity to greater values will violate the comfort zone and cause discomfort long before pixel averaging artifacts become noticeable. In Sec. 3.5, we describe a perceptual study, in which optimal target disparities are found.

Furthermore, as we mentioned before, SPSS is only suitable for scenes with planar depth structure. Since planar structures can be linearly modeled, their depth can be accurately enhanced using linear remapping. We do not apply SPSS on non-planar structures since visual distortion can happen. For example players may look tilted towards the ground. However, since most of the shots in field sports have a planar structure, our SPSS method can improve depth for a significant portion of shots. In Sec. 3.5.4, we analyze the coverage of our method and show that 60–70% of the shots can benefit from SPSS.

### 3.4.2 Depth Personalization

The perception of depth varies from person to person, as it is a matter of personal preference. Thus, using the same content in all situations is sub-optimal, both in terms of depth perception and visual comfort. In this section, we describe a depth personalization method that allows a viewer to choose his/her preferred depth perception level. Depth personalization is realized as follows. The user interface at the client side displays multiple depth options. The default selection is chosen based on the viewer’s display size. However, the viewer is allowed to choose other versions with more or less depth. After selection, the client translates this request to the corresponding segment ID and submits it to the DASH server, which streams the requested segments.
In order to achieve depth personalization, we perform three main operations: (i) Create multiple versions with different depth values, (ii) Control the popping-out effect, and (iii) Integrate versions with DASH architecture. These operations are discussed in the following.

**Creating Multiple Versions with Different Depth Values:** We start by determining the upper and lower bounds at which the depth can be perceived. The upper bound is the maximum depth that can be fused by the human visual system. The lower bound is the depth at which no object appears in or out of the display (i.e., almost like 2D). As shown in Fig. 2.2, the amount of perceived depth \( z \) is defined by \( z = \frac{p}{e/d - 1} \), where \( d \) is the disparity between the left and right images, and \( e \) is the inter-ocular distance. Therefore, the point where \( z \) has the maximum value happens when \( d = e \) [86]. This is the point where \( z = \infty \). On the other hand, the point where there is no depth perception \( (z = 0) \) happens when \( d = 0 \). We adjust the target disparity \( \hat{d} \) using the upper and lower bounds on disparity. We set \( d_{\text{max}} = e \) for maximum depth expansion, and \( d_{\text{min}} = 0 \) for maximum depth compression. To convert \( \hat{d} \) to pixels we should multiply it by the pixel density of the display, where pixel density is the diagonal resolution of the display in pixels, divided by the length of the diagonal. The maximum disparity can then be expressed in pixels as:

\[
\hat{d}_{\text{max}} = \text{pixel\_density} \times e,
\]

where \( e \) is a constant value equal to 2.5 inches, which is the typical distance between human eyes. If the disparity exceeds this amount the human visual system will not be able to fuse the two views.
Figure 3.8: Our system provides users with three options for controlling the popping out effect: (a) Increasing pop out effect, (b) Increasing depth range while not increasing the pop out effect, and (c) Removing pop out effect.

The above formula suggests that the maximum disparity allowed on a display depends on the pixel density of that display. This means that in order to have an effective depth personalization, the maximum disparity presented to a tablet user should be different than the one presented to a TV user. For example, a HTC Evo 3D mobile phone has a pixel density of 256 pixels per inch, and therefore the $\hat{d}_{\text{max}}$ for this phone is 640 pixels. On the other hand, the pixel density for a 55" Philips TV-set is 40 pixels per inch, and therefore the $\hat{d}_{\text{max}}$ of this TV-set is only 100 pixels.

Next, using our depth customization method in Sec. 3.4.1, we create multiple versions between the minimum and maximum depth values. Each of these versions represents a different level of depth. For simplicity of implementation, we create the versions (20 in our case) at equal distance in the range of ($\hat{d}_{\text{min}}, \hat{d}_{\text{max}}$). For $\hat{d}_{\text{max}}$, we choose the largest $\hat{d}_{\text{max}}$ among all supported devices. All versions are stored in a manifest file that contains meta data about the video and its available versions. Each version has an attribute in the manifest file called target_disparity, which is the $\hat{d}$ value used to compute the version. This value is used to prevent clients from requesting versions that exceed the maximum allowed disparity on their displays. Fig. 3.7 shows some samples of expanded and compressed depths for the sample image in Fig. 3.4a. The figures from left to right show how the depth range gradually increases so that it can match various users’ preferences.

**Controlling Popping-out Effects:** Objects that pop out of the display create a nice, catchy effect for many viewers. However, if the popping out is excessive or not properly adjusted, it can cause visual discomfort and often headaches [109, 107]. In order to manage the trade-off between creating catchy effects and maintaining visual comfort, our system provides the user with three options to control the popping out effects, as follows:

1. **Increasing pop out effect:** This is the default case for our method and it is done by setting $\beta = 0$ in Eq. (3.1). Applying this equation expands the depth in both directions, meaning that the objects that are originally popping out of the display pop out more, and the objects originally inside the display go deeper inside. (see Fig. 3.8a)

2. **Increasing depth range while not increasing the pop out effect:** In this case, the scene is adjusted in a way that the objects that used to pop-out in the original scene will maintain the same depth perception and the objects inside the display plane go deeper inside. This
avoids any discomfort caused by having some objects strongly popping out of the display and at the same time it maintains the catchy effects that appear in the original scene. Hence, we set the pop-out factor in a way that the scene is pushed back inside until it reaches the maximum negative disparity of the original scene. In Eq. (3.2), it is shown that the maximum negative disparity added to the original image is given by \[ -\frac{|\sigma_l|+|\sigma_u|}{2}. \] Therefore, we set \[ \beta = \frac{|\sigma_l|+|\sigma_u|}{2} \] to cancel the added negative disparity. Since the value of \( \beta \) is added uniformly to all pixels, such adjustment does not affect the expanded depth range achieved by Eq. (3.1). It however, ensures that the maximum negative disparity is maintained at the original value. (see Fig. 3.8b)

3. Removing pop out effect: Some viewers do not like the pop out effect. For such viewers, we adjust the scene so that all objects are perceived behind the display plane. In order to do this, we set the pop-out factor in a way that the pixels with maximum negative disparity are shifted to be on the display plane. Therefore, we set \[ \beta = \frac{|\sigma_l|+|\sigma_u|}{2} + |d_{\text{max,negative}}|, \] where \( d_{\text{max,negative}} \) is the maximum negative disparity in the original scene. This value for \( \beta \) removes all negative disparities in the scene. (see Fig. 3.8c)

Integrating Versions with DASH Architecture: DASH is an adaptive streaming protocol that uses HTTP for streaming purposes. Videos under DASH are divided into small segments and encoded with different bitrates. This enables DASH clients to adapt to variable network conditions smoothly. Further, DASH defines a manifest file called Media Presentation Description (MPD) that contains the available segments bitrates, codecs, resolutions and timing information. Typically, DASH client requests MPD and parses its content. When the streaming session starts, it sends HTTP request to the server with the chosen segment, and the server replies back with the corresponding segment. Our client first requests the manifest file from the server and decides on the most suitable 3D version offered by the server based on current network conditions and display characteristics. The client then starts requesting video segments from the server. During the video playback, the viewer is allowed to manually expand/compress the depth of the video. If the user decides to expand the depth of the video, the client looks up the manifest file and requests the next segment from the version with the expanded depth perception. However, if the \text{target_disparity} attribute of the requested version is greater than the display’s \( \hat{d}_{\text{max}} \), the client will not switch to the expanded version and the user will be notified that the current version has the maximum depth suitable for his/her display. In case the \text{target_disparity} attribute of the requested version is less than the display’s \( \hat{d}_{\text{max}} \), the client continues fetching subsequent segments from the expanded version. At any point of time, the user can return back to the previous setting by choosing to compress the depth, and similarly, the user will be notified when the current version has the most compressed depth. Once the user is satisfied with the depth perception, he/she is allowed to adjust the \text{pop-up factor} to one of the three options described earlier.
3.5 Evaluation

We conduct subjective studies to evaluate the depth quality in 3D videos. Our subjective studies show that up to 25% improvement in the depth quality can be achieved by the proposed system.

We start our evaluation by showing the need for depth customization and personalization through subjective studies. Next, we analyze the impact of depth personalization, and show that increasing the depth level improves depth perception, but can cause visual discomfort if performed excessively. We then measure the improvements achieved by our system for different viewing conditions. Finally, we analyze two full soccer games and a 10-min tennis game and show that our method can enhance between 60% to 70% of the shots in 3D videos of field sports, such as soccer and tennis, while keeping the rest of the shots unchanged.

3.5.1 The Need for Depth Customization and Personalization

In this study, we manipulated the depth of a 3D video with different degrees and displayed them to multiple subjects to discover their preferences. We conducted two experiments. In the first experiment, we focus on the need of depth customization for different displays and viewing conditions. In the second experiment, we focus on the need for depth personalization to satisfy the preference of different viewers. We note that early works, e.g., ATTEST [102], showed the need for depth customization, but for a specific type of experimental displays and video plus depth format. Our experiments show this need for various modern, commodity, 3D displays including mobile phones and active and passive TV displays.

**Setup:** We used a series of short clips taken from the Manchester United vs. Wigan Athletic FA Community Shield game (2–0, 12 August, 2013). Each clip was a few seconds long, and the entire sequence was 54 seconds long. The initial depth range of the test sequence was small, and the whole image appeared flat. The subjects were shown six versions of the test sequence, with increasing expansion values. The values used were: $-0.5\%$ (depth compression), $0\%$ (original sequence), $1\%$, $2\%$, $3\%$, and $4\%$ (depth expansion). In each condition, the subjects were asked to choose the preferred version, assuming that they were to watch the entire 90-minute soccer game.

**Depth Customization:** Ten subjects participated in this study. They were all computer science students and researchers, and all could perceive stereoscopic 3D effect. We tested two displays: smartphone and TV. The smartphone model was LG Optimus 3D MAX P725 in which the stereo effect is achieved by means of a parallax barrier. The observation distance was about 30 cm. The TV was a Sony Bravia XBR-55HX929 TV-set, and a pair of active shutter glasses. The observation distance was about 2 m. Half of the subjects saw the TV condition first, and the other half saw the smartphone condition first.

The results together with the standard error of the mean (confidence interval = 68%) are shown in Fig. 3.9. Results show that the average preferred values for expansion were: $1.4\%$ for the TV and $2.3\%$ for the smartphone. The experiment shows that the original 3D videos were not the best choice for subjects. Instead, the depth-manipulated 3D videos scored higher than the original
ones. In addition, the level of depth manipulation depended on the used display. Therefore, we can conclude that for optimized viewing of 3D videos on different displays, the depth of the videos needs to be adjusted.

We use the results of this experiment to set the default expansion values. That is, the expansion values for the TV and smartphone are set to 1.4% and 2.3%, respectively. We use linear interpolation of the expansion value for the other display sizes used in the experiments in Sec. 3.5.3.

![Figure 3.9: Mean expansion value and standard error of the mean for two displays.](image)

![Figure 3.10: Histogram of the expansion values preferred by subjects.](image)

**Depth Personalization:** For this experiment, we enlarged the number of subjects to capture various personal differences. Twenty five subjects participated in this study. They were all computer science students and researchers, and all could perceive stereoscopic 3D effect. We used a 4K TV display (59.5" LG 60UF8500) with a viewing distance of about 3 m. The results (Fig. 3.10) show the diverse depth preferences of the subjects.
3.5.2 Impact of Depth Personalization

The depth personalization feature of our system allows users to adjust depth based on their own preferences by increasing/decreasing the depth level. In this study we assess the impact of different depth levels on the 3D video perception. According to the ITU BT.2021 recommendations [16], there are three primary perceptual dimensions for 3D video assessment: picture quality, depth quality and visual (dis)comfort. Picture quality is mainly affected by encoding and/or transmission. Depth quality measures the amount of perceived depth, and visual discomfort measures any form of physiological unpleasantness due to 3D perception, i.e., fatigue, eye-strain, head-ache, and so on. Such discomforts often occur due to 3D artifacts, depth alteration, comfort zone violations and/or cross talk. In this study, we measure depth quality and visual comfort. We do not measure picture quality because we do not change any compression or encoding parameters.

We display sequences indoors on a 4K 59.5" LG TV-set with passive polarized glasses, in low lighting conditions. The viewing distance was around 3m. Fifteen subjects took part in this experiment. They were all computer science students and researchers. Their stereoscopic vision was tested prior to the experiment using static and dynamic random dot stereograms.

We used three soccer 3D video clips. From each clip, we used a series of shots with the following total lengths. (i) Manchester United vs. Wigan: 60 sec. (ii) Chelsea vs. Wigan: 24 sec. (iii) Chelsea vs. Plymouth: 20 sec. All shots were of long-view nature, thus suitable for our method. In our system, these shots are automatically identified by a classifier. Other shots are not subjected to our depth customization. The subjects viewed five versions of each sequence: the original version (level 0) and four versions with higher depth levels provided by our system. For all versions, $\beta$ (the pop-out factor) was set to the default value of zero.

We use the single-stimulus method of the ITU recommendations to assess depth quality and visual comfort. The sequences are shown to subjects in random order. As suggested by the ITU recommendations, each sequence is preceded by a 3 sec mid-grey field indicating the coded name of the sequence, followed by a 10 sec mid-grey field asking subjects to vote. We use a discrete five-grade scale to rate depth quality and comfort. The depth quality labels are 5–Excellent, 4–Good, 3–Fair, 2–Poor, and 1–Bad, while the comfort labels are 5–Very Comfortable, 4–Comfortable, 3–Mildly Uncomfortable, 2–Uncomfortable, and 1–Extremely Uncomfortable. The subjects are asked to rate the amount of depth they can perceive in each sequence along with how comfortable it was to watch the sequence. We asked subjects to clarify all their questions and ensure their full understanding of the experimental procedure.

The inclusion of the original sequence allows us to compute the Difference Opinion Score (= score of each sequence - score of the original sequence). We then calculate the mean of the difference opinion scores (DMOS). A DMOS of zero implies that the sequence is judged the same as the original one, while a negative DMOS implies a depth perception/comfort lower than the original.

Fig. 3.11 shows the DMOS values along with the standard error of the mean for a confidence interval of 68%. The figure shows that for the first few levels, increasing the depth level enhances
depth perception without any noticeable degradation in comfort (level1, level2). However, when the depth exceeds a certain amount, visual comfort will drop dramatically (level4). In other words, increasing the depth level has a positive impact on the 3D quality, but if increased excessively it will cause degradation in visual comfort and thus the 3D quality. Since the optimal depth is highly subjective, depth personalization enables users to choose the level that maximizes their depth perception while being comfortable to watch the content.

### 3.5.3 Depth Improvement

We conduct a subjective study to measure the perceptual depth enhancements achieved by our system for different videos and viewing conditions. We use the double-stimulus method for this experiment. Subjects view each pair of sequences at least twice before voting so as to assess their differences properly. The sequences are shown in random order without the subjects knowing which is which. We use the three video sequences in Sec. 3.5.2 and display them under 5 different viewing conditions: (i) Smartphone: LG Optimus 3D MAX P725 phone (autostereoscopic), with observation distance about 30 cm. (ii) Tablet: GADMEI tablet (autostereoscopic), with observation distance
Figure 3.12: Difference Mean Opinion Score (DMOS) between our optimized version and the original 3D sequence. Our method improves depth quality for all tested videos on all displays. Error bars represent the standard error of the mean.

about 40 cm. (iii) Laptop: 15.6" Toshiba Qosmio F755-3D350 laptop (autostereoscopic with eye tracking), with observation distance about 60 cm. (iv) Desktop: 27" Samsung desktop (active shutter glasses), with observation distance about 1 m. (v) Big TV: 55" Philips TV-set (passive polarized glasses), with observation distance about 3 m. Fifteen subjects took part in this experiment. All subjects viewed the five 3D displays. For each display, the subjects were shown the original sequence and the optimized version. Both the order in which participants saw the displays and the video sequences were randomized. The subjects were asked to rate the depth quality of each version on a five-grade discrete scale as follows: 5–Excellent, 4–Very Good, 3–Good, 2–Fair, and 1–Poor.

As suggested by ITU, we compute and report the DMOS values. The DMOS results for all 3D videos are shown in Fig. 3.12. The figure shows the difference between ranking values of the depth-optimized 3D videos created by our method and the original 3D videos. The error bars visualize the standard error of the mean with confidence interval of 68%. The results demonstrate that our method improves the depth quality in all tested cases.

We note that the current stereo videos for sports are shot with stereo cameras with fixed parameters, and typically these parameters are set conservatively so that the produced 3D video can be viewed on many displays. In other words, typical depth ranges in original 3D videos are small, which leaves large room for improvements for our method. For example, Fig. 3.12a shows that the
MOS of the depth quality for TV improved by up to 1.4 points compared to the original sequence. This means that more than 25% improvement was achieved using our method.

Finally, we checked the statistical significance of our results by performing the Wilcoxon signed-rank test. All differences reported are statistically significant (p-value $< 0.05$) except the tablet version of the second video (Fig. 3.12b), and the desktop and laptop versions of the third video (Fig. 3.12c).

### 3.5.4 Coverage of the Depth Enhancement Method

To demonstrate the percentage of shots that can benefit from our depth customization method, we analyze two full 3D soccer games (each is more than 90 minutes), and a 10-min segment of 3D tennis game. The two soccer games were from different broadcasters and different competitions. We watched each video and manually classified shots. We summarized our analysis of the first full 3D soccer game, Milan vs. Barcelona, in Table 3.1, which shows the percentage of long, medium, close-ups, and out-of-field shots. Similar to any usual soccer game, the game included goals, player changes, injuries, offsides, etc. It can be seen that the percentage of shots is similar throughout the two halves of the game. During the second half, two player changes, two injuries, and two goals happened, which decreased the percentage of long shots. This is because these events typically have more close-up shots. However, on average, over 70% of the full 3D soccer game is long shots. Table 3.2 shows the analysis of another full 3D soccer game (Manchester United vs. Liverpool). The main difference between the two games is the percentage of close-up shots. Nevertheless, the shot percentage is still dominated by long shots in both games.

In Table 3.3 we summarized our analysis of the 3D segment from the tennis game between Rafael Nadal and Novak Djokovic. It can be seen that our method can enhance more than 64% of the game.

<table>
<thead>
<tr>
<th>Time/Shot Type</th>
<th>Long</th>
<th>Medium</th>
<th>Close-up</th>
<th>Out-of-field</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00 - 00:02:10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introduction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:02:10 - 00:48:17</td>
<td>73.1%</td>
<td>24.4%</td>
<td>1.3%</td>
<td>1.2%</td>
</tr>
<tr>
<td>00:48:17 - 1:03:20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Half-time</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1:03:20 - 1:53:52</td>
<td>67.1%</td>
<td>28.4%</td>
<td>1.4%</td>
<td>3.1%</td>
</tr>
</tbody>
</table>

Table 3.1: Shot analysis of the Milan vs. Barcelona full 3D soccer game on 20/2/2013 in the UEFA Champions League.
<table>
<thead>
<tr>
<th>Time/Shot Type</th>
<th>Long</th>
<th>Medium</th>
<th>Close-up</th>
<th>Out-of-field</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00 - 00:04:30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:04:30 - 00:55:10</td>
<td>68.25%</td>
<td>26.25%</td>
<td>4.1%</td>
<td>1.4%</td>
</tr>
<tr>
<td>00:55:10 - 1:06:56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1:06:56 - 1:58:10</td>
<td>61.3%</td>
<td>29.4%</td>
<td>6.6%</td>
<td>2.7%</td>
</tr>
</tbody>
</table>

Table 3.2: Shot analysis of the Manchester United vs. Liverpool full 3D soccer game on 16/3/2014 in the English Premiere League.

<table>
<thead>
<tr>
<th>Time/Shot Type</th>
<th>Long</th>
<th>Medium</th>
<th>Close-up</th>
<th>Out-of-field</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00 - 00:00:28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:00:28 - 00:09:41</td>
<td>64.4%</td>
<td>16.8%</td>
<td>2%</td>
<td>16.8%</td>
</tr>
</tbody>
</table>

Table 3.3: Shot analysis of the R. Nadal vs. N. Djokovic 3D tennis clip on 3/7/2011 in the Wimbledon competition.

In summary, our analysis of various 3D games shows that our method can enhance between 60% to 70% of the shots in 3D videos in field sports, such as soccer and tennis.

### 3.5.5 Running Times

In this experiment, we measure the processing time of each operation implemented in the system. We measure the time on one server with Intel Xeon CPU E5620 (2.4 GHz), 12 GBs of memory and Nvidia Tesla C2075 GPU with 6 GB GPU memory. All operations are implemented using C++ and OpenCV. Figure 3.13 shows the time required to perform each operation. The figure shows that most of the operations’ processing time is below real time which is 42 milliseconds (assuming 24 fps). The anaglyph operation is slightly higher than real time because it requires upscaling the image to double the resolution. This operation takes 70% of the time needed to run the anaglyph operation. Nevertheless, with some slight optimizations the anaglyph operation can run in realtime.

In order to speed up the computation, we use GPU to run our depth optimization operation. The figure shows that the total running time for this operation is 245 milliseconds with standard deviation of ±5 milliseconds. Although, the operation runs in 6 × realtime it is much faster than other methods that rely on creating the depth map before customizing the depth perception. In order to create the depth map in a good quality it takes from 10 seconds to 2 minutes for each frame, which is at least 40 times slower than our method.

### 3.6 Summary

In this chapter, we presented a novel system for adaptive streaming of stereoscopic content. The key feature of our solution is the capability of providing high quality stereoscopic 3D videos to het-
Figure 3.13: Time required to process each operation in our system. The horizontal line is at $t = 42$ msec, representing real time processing at 24 fps.

erogeneous receivers in terms of display sizes and type, viewing conditions, viewers’ preferences, and network conditions. To support such capabilities, our system employs the DASH protocol for adaptive streaming and switching among different versions of 3D videos. In addition, we proposed a new method for depth customization of 3D sports videos such as soccer, football, and tennis. The technique is computationally inexpensive and maintains the scene structure. Together with the adaptability of DASH streaming, it enables depth personalization to allow users to adjust the depth of a video based on their own preferences. This is a key feature of our system, which not only improves the viewers’ engagement but also provides a comfortable experience. To evaluate the performance of our system, we conducted a series of user experiments with different content and display devices showing that our technique can significantly improve the perceived quality.
Chapter 4

Gradient-based 2D-to-3D Conversion for Sports Videos

4.1 Introduction

Stereoscopic 3D (S3D) videos offer more engaging experience to viewers than traditional 2D videos, especially for sports games. Shooting sports games in 3D, however, is complex and costly, because it requires deploying and operating expensive 3D camera rigs. A more cost-effective approach is to convert regular 2D videos to 3D using automated methods. The 2D-to-3D conversion methods can also be used to convert previous events of historical importance, e.g., the previous FIFA World Cup final game. Converting 2D sports videos to high-quality 3D is, however, challenging, because of the high motion and complexity of the scenes in sports games. Current 2D-to-3D conversion methods, e.g., [94, 135], are designed for general videos and when applied to sports videos may introduce various visual artifacts that negatively impact the viewing experience of users.

In this chapter, we propose a data-driven method for converting soccer 2D videos to 3D. The proposed method handles the temporal and spatial complexities of soccer videos. Unlike several previous methods, e.g., [74, 71], the proposed method is designed and optimized for sports videos and especially soccer videos. The key idea of the proposed method is to learn the depth information of a video frame from similar frames in a database of 2D+Depth soccer images. However, such databases are very costly to create, especially for outdoors sports games where depth information is harder to capture compared to indoor environments where simpler equipment (e.g., Microsoft Kinect) can be used to capture depth. In addition, sports games may contain numerous varieties of scenes and frame compositions, which requires large and diverse databases to cover. We address this problem by creating a synthetic database from computer games and showing that this synthetic database can effectively be used to convert real videos. Current computer games provide high-quality depth maps, which allows us to cost-effectively obtain a wide variety of shots from different teams, stadiums, seasons and camera angles.
The proposed method converts individual frames by dividing each into blocks and finding similar blocks in the database. It then transfers the depth gradient from the matched blocks. This, however, may not produce smooth depth within the frame and across successive frames. We present a spatio-temporal depth reconstruction method to address this problem.

We conduct extensive user studies to evaluate the performance of the proposed 2D-to-3D conversion method. In these studies, we use a diverse set of video segments and follow the ITU BT.2021 recommendations [16]. Our results show that: (i) 3D videos produced by our method are almost indistinguishable from original videos shot in 3D, (ii) our converted videos are rated Excellent by subjects, most of the time, and (iii) our method significantly outperforms the state-of-the-art method in the literature [71].

In this chapter, we first summarize the related works in the literature in Section 4.2. We then provide an overview of our proposed method in Section 4.3. Section 4.4 presents the proposed depth gradient based conversion method and Section 4.5 presents the object segmentation method. Section 4.6 describes our subjective and objective evaluation, and Section 4.7 summarizes the chapter.

### 4.2 Related Work

Over the last few years, applications for 3D media have extended far beyond cinema and have become a significant interest to many researchers. Liu et al. [83] discuss 3D cinematography principles and their importance even for non-cinema 3D content. Wu et al. [126] adapt 3D content quality for tele-immersive applications in real-time. Calagari et al. [30] propose a 3D streaming system that performs depth customization for a wide variety of 3D displays. Yang et al. [129] use the client viewing angle in a tele-immersive environment to prioritize the streaming of 3D content. Hefeeda et al. [61] provide content protection for 3D media. While such systems provide useful applications, the limited 3D content still remains a main bottleneck for the adoption of 3D technology. To tackle this issue, 2D-to-3D conversion techniques can be used. 2D-to-3D conversion has been explored by many researchers. However, previous methods are either semi-automatic [100, 136] or cannot handle complex motions [94, 73, 106, 74, 62, 71]. To the best of our knowledge, there has not been a 2D-to-3D conversion technique that is capable of handling the complex motions and the variety of scene structures that exist in soccer videos.

In 2D-to-3D conversion, the depth map of an image is estimated. Stereo image pairs can then be synthesised using this depth information. Depth maps can be computed using traditional computer vision approaches such as structure from motion or depth from defocus. Park et al. [94] estimate the depth using structure from motion. Zhang et al. [136] use multiple depth cues including defocus and motion for their proposed semi-automatic 2D-to-3D conversion system. A survey on automatic 2D-to-3D conversion techniques and depth cues can be found in [133]. In several of the previous works, strong assumptions are often made on the depth distribution within a given scene. For example, the work in [73] classifies shots into long shots and other shots (e.g., medium shots, close-ups, bird’s eye view, etc.), where long shots are the shots with a large field view. Long shots are assigned a
depth ramp for the field and a constant depth for the players. Similarly in [106], players are detected and a constant depth is assigned to them. This, however, causes the well-known ‘card-board effect’ where supposedly 3D objects appear flat on the screen.

Data-driven methods are an alternative way of computing depth maps. A relatively coarse depth estimation is provided in Hoiem et al. [62], where a scene is segmented into planar regions, and an orientation is assigned to each region. Konrad et al. [74] use a database of image and depth map pairs to infer the depth of an input image. Their work is designed for still images and assumes that images with similar gradient-based features tend to have a similar depth. For a query image, the depth is estimated as the median over the depths of the most similar images from the database. Karsch et al. [71] extended this approach to image sequences. While also using a large database of image and depth map pairs, the most similar images in the database are found and warped to the query image. The final depth map is then estimated by combining the warped images. The work in [71] is the closest to ours and we compare against it. There are a few commercial products that provide automated 2D-to-3D conversion, sold as stand-alone boxes (e.g., JVC’s IF-2D3D1 Stereoscopic Image Processor, 3D Bee), or software packages (e.g., DDD’s TriDef 3D). While the details of these systems are not known publically, their depth quality is still an outstanding issue [133].

4.3 System Overview

An overview of the proposed 2D-to-3D conversion method is shown in Fig. 4.1. We infer the depth from a database of synthetically generated depths. We collect this database from video games. With the high quality of current video games, which has come close to that of real videos, using a synthetic database offers two main advantages: 1) we can obtain a diverse database from different camera
angles, teams, and stadiums; and 2) we can obtain accurate depth maps with perfect discontinuities. We discuss our synthetic database in Sec. 4.4.1.

For each query image, we transfer the depth gradients from the synthetic database to the query image by dividing the query into blocks and copying the depth gradients from the matching blocks in the database. This is quite different from previous approaches that use absolute depth over the whole frame [74, 71]. Our approach offers finer depth assignment to smaller objects (e.g., players), while requiring a much smaller database. This is because we match small blocks instead of the whole frame, and blocks have much less variety than frames.

After the depth gradients have been transferred, we recover the depth from these gradients by using Poisson reconstruction. Poisson reconstruction is a robust technique traditionally used to recover an image from its gradient information by solving a Poisson equation [97, 22]. We enhance the Poisson reconstruction formulation such that it utilizes temporal gradients in addition to spatial gradients. Our spatio-temporal Poisson reconstruction enables the generation of temporally smooth depth maps. Our depth estimation technique is discussed in Sec. 4.4.

In order to maintain clear object boundaries, we create object masks and allow depth discontinuities on object boundaries by modifying the Poisson equation. We present two different methods for creating object masks, one for close-up shots and the other for non close-ups. In order to distinguish these two types of shots, we implemented a simple shot classification method. Sec. 4.5 discusses our object mask creation methods.

Finally, we use the stereo-warping technique in [71] to render the left and right stereo pairs using the 2D frames and their estimated depth. In this technique, a 2D frame is warped based on its estimated depth such that salient regions remain unmodified, while background areas are stretched to fill dis-occluded regions.

### 4.4 Gradient-based Conversion

The core of our system is depth estimation from depth gradients; for an input 2D video, depth is inferred from our synthetic database. Fig. 4.2 outlines this process. For a 2D query frame, we first search the database for the $K$ nearest frames. Using these $K$ candidates we create a matching image block by block, where for each block we choose the best matching block from the $K$ candidates. We then copy the depth gradients from the matched blocks to the query frame. Finally the depth is reconstructed from these copied gradients by solving a Poisson equation. We now discuss each step in more detail.
Figure 4.3: The effect of different steps in the proposed depth estimation method: (a) Query, (b) Subset of \( K \) matching candidates, (c) Created matched image, (d) Object boundary cuts, (e) Depth estimation using Poisson reconstruction, (f) Gradient refinement and Poisson reconstruction, (g) Depth with object boundary cuts, (h) Final depth estimation with smoothness, and (i) A zoomed and amplified version of the yellow block in h.

4.4.1 Synthetic Database

Many databases of RGBD (Red, Green, Blue and Depth) images [7, 3, 12] and videos [71, 9] have been created. The depth channel is acquired using time-of-flight imaging [104] or active stereo (e.g., using Microsoft Kinect). However, none of the current RGBD databases can be used for sports events. Acquiring depth maps for sports events is challenging since it requires the depth to be captured in sunlight conditions and in a highly dynamic environment. In order to address this challenge, we propose to create a Synthetic RGBD (S-RGBD) database from video games. Current video games have very high image quality and a large quantity of content can be easily generated from them.

To collect our S-RGBD data we use PIX [11], a Microsoft Directx tool, to extract image and depth information from the FIFA13 video game. PIX records all Directx commands called by an application. Each recorded frame can then be rendered and saved by re-running these commands. In addition, PIX allows access to the depth buffer of each rendered frame. We extracted 16,500 2D+Depth frames from 40 different sequences. Each sequence has a frame rate of about 10 fps, and each extracted frame has a resolution of \( 1916 \times 1054 \). These 40 sequences cover a wide range of shots that can occur in a soccer match, including a variety of camera angles, color variations and motion complexities. Two of the 40 sequences are designed to capture the common scenes...
throughout a full game. Each one has a duration of $6 - 7$ minutes. The rest of the sequences are shorter ($15 - 60$ seconds) and focus on capturing special and less common events such as behind the goal, close-ups, and zoomed on ground views. Our database includes different stadiums, teams, seasons and camera views.

### 4.4.2 Block-based Matching

For each frame of the examined video we first identify the $K$ ($= 10$ in our work) most similar frames in our S-RGBD database by performing visual search. The two main features used for visual search are: GIST [89] and Color. The former favors matches with overall similar structure, while the latter favors matches with overall similar color. For color, we use the hue channel in the HSV color space and create a normalized histogram of hue values with six equal-width bins. We then apply a binary thresholding with value 0.1 to represent only dominant colors. We then concatenate GIST and the color histogram to form the final image search descriptor. Fig. 4.3(b) shows $K$ candidates for the frame in Fig. 4.3(a).

Using the $K$ candidate images we construct a matched image, which is an image similar to the examined frame. The matched image provides a mapping between the examined frame and the candidates, where each pixel in the examined frame is mapped to a corresponding candidate pixel. While such mapping can be performed using a global approach by warping the candidates to the examined frame, such as [71], this requires strong similarity between the examined frame and the database. For example, an examined frame with 4 players requires the database to have an similar image. Therefore, we use a local approach instead, where similar images are constructed using block matching. This provides a more robust matching. For example, a good matching can be performed between two images even if they are shot from different angles and different locations, and have a different number of players. This can be seen in the example in Fig. 4.3 where the images in Fig. 4.3(b) were used to create the high-quality matched image in Fig. 4.3(c), which may not have been possible using the global approach in [71]. One of the advantages of our local approach is that it can achieve good results without requiring a massive database, which is highly desirable since, as discussed in Sec. 4.4.1, creating an accurate 3D database is difficult.

For constructing the matching image, the examined frame is first divided into $n \times n$ blocks. In all of our experiments, we set $n$ to 9 pixels. Each block of the examined frame is then compared against all blocks in the $K$ candidate images. We compare blocks based on their block descriptors. The block whose block descriptor has the least Euclidean distance with that of the examined block is chosen as the corresponding block. For block descriptor, we concatenate the SIFT descriptor calculated for the center of the block with the average RGB value of the block. Note that the candidate images are all re-sized to the examined frame size. RGB values are normalized between 0-1. To capture more representative texture, the SIFT descriptor for each block is calculated on a larger patch of size $5n \times 5n$. Fig. 4.3(c) shows the matched image using our block matching approach. Notice that the horizontal playing field is matched with the horizontal playing field, the
vertical advertisement boards are all matched with vertical blocks, and the tilted audience are also matched with the audience.

Note that for a faster matching, we process the image search descriptors, the block descriptors, and the depth gradients for all frames in the database beforehand and store them as the database. Therefore, in practice, there is no need to actually store the RGB frames and depth maps of the database frames, which saves a considerable amount of storage, in addition to reducing the processing time.

### 4.4.3 Poisson Depth Estimation

**Computing Depth Gradients:** Given a query frame and its matched image, we copy the corresponding depth gradients in blocks of \( n \times n \) pixels from the matched image to the query frame. By depth gradients we refer to the first order spatial derivatives of the depth for both horizontal and vertical directions (\( G_x, G_y \)).

**Poisson Reconstruction:** We reconstruct the depth values from the copied depth gradients using the Poisson equation:

\[
\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) D = \nabla \cdot G,
\]

where \( G = (G_x, G_y) \) is the copied depth gradient and \( D \) is the depth we seek to estimate. \( \nabla \cdot G \) is the divergence of \( G \):

\[
\nabla \cdot G = \left(\frac{\partial G_x}{\partial x} + \frac{\partial G_y}{\partial y}\right).
\]

In the discrete domain, Eq. (4.1) and Eq. (4.2) become Eq. (4.3) and Eq. (4.4), respectively:

\[
D(i, j + 1) + D(i, j - 1) - 4D(i, j) + \nabla \cdot G(i, j).
\]

\[
\nabla \cdot G(i, j) = G_x(i, j) - G_x(i, j - 1) + G_y(i, j) - G_y(i - 1, j).
\]

To estimate the \( D \) of each pixel, we formulate the problem in the form of \( Ax = b \), where \( b \) is a vector containing the \( \nabla \cdot G \) values for all pixels, \( x \) is a vector containing the depth values (\( D \)) for all pixels, and \( A \) is a matrix storing the coefficients of the Poisson equation (Eq. (4.3)). Note that in this thesis, all vectors and matrices are denoted as bold letters. For a query image of size \( H \times W \), \( A \) is a square matrix with size \( HW \times HW \). Each row in \( A \) corresponds to a pixel in the query frame, and the values in the row correspond to the coefficients of Eq. (4.3). Fig. 4.4 illustrates setting up \( A \) for a small sample image. Note that since one or more neighbors do not exist for the image boundary pixels, the value of \( \nabla \cdot G \) in these pixels is updated by removing the terms in Eq. (4.4) that refer to non-existing neighbors. Finally, given \( Ax = b \), we solve for \( x \). An example of the
reconstructed depth ($x$) is shown in Fig. 4.3(e). In can be seen that the overall depth structure is captured, however, there are some artifacts present (see the lower right corner of Fig. 4.3(e)). Such artifacts are often caused by the inaccuracy in SIFT matching. For example, in Fig. 4.3(c) some field blocks are matched to non-field areas. If a query block from a region that is expected to have smooth depth (such as the field) is incorrectly matched to a reference block that has sharp changes in depth (such as player borders or the goal), small artifacts in the depth map can occur due to the sharp gradients that were transferred from the reference block. To avoid this problem, we perform gradient refinement, which reduces the large gradients before solving for $x$. Then using our object masks we impose depth discontinuities in the proper places. We describe these two steps in the following.

**Gradient Refinement:** To reduce the errors caused by incorrect block matchings, we multiply the depth gradients by a refinement factor:

\[
G_x = G_x \times \max(1 - e^{\frac{1}{\alpha |G_x|}}, 0)
\]

\[
G_y = G_y \times \max(1 - e^{\frac{1}{\alpha |G_y|}}, 0)
\]

(4.5)

This refinement exponentially reduces large gradients, which may be incorrectly estimated, while maintaining low gradients. The refinement strength is configured by the parameter $\alpha$. A high $\alpha$ can corrupt correct gradients, while a low $\alpha$ can allow artifacts. In all our experiments, we set $\alpha$ to 60. Fig. 4.5 shows the refinement factor for $G_x$ when $\alpha$ is set to 60. It can be seen that while the factor is 1 for small values of $G_x$, it drops to zero as the gradient starts to grow. Fig. 4.3(f) shows the effect
Figure 4.5: The refinement factor of $G_x$ for $\alpha = 60$.

of gradient refinement on depth estimation for Fig. 4.3(a). In comparison to Fig. 4.3(e), artifacts are removed and depth is smoother.

Object Boundary Cuts: When performing Poisson reconstruction each pixel is connected to all its neighbors. This causes fading of most object boundaries, especially after gradient refinement where strong gradients are eliminated (see Fig. 4.3(f)). We solve this issue by modifying the Poisson equation on object boundaries and allowing depth discontinuities. To do so, we use object masks, which are discussed in Sec. 4.5. Given object masks, we first use the Canny edge detector to detect edges (see Fig. 4.3(d)). We then disconnect pixels from the object boundaries by preventing them from using an object boundary pixel as a valid neighbor. For each pixel neighboring a boundary pixel, the corresponding connection in $A$ is set to 0 and its $\nabla \cdot G$ value is updated accordingly. Hence, pixels adjacent to object boundaries are treated similar to image boundary pixels.

The object boundaries generated for Fig. 4.3(a) are shown in Fig. 4.3(d). The estimated depth when cutting the object boundaries is shown in Fig. 4.3(g). The players in Fig. 4.3(g) are more visible compared to Fig. 4.3(f).

Smoothness: For a smoother depth, smoothness constraints are added to the Poisson reconstruction. These constraints enforce the higher-order depth derivatives to be zero. In continuous domain we set

$$
\left( \frac{\partial^4}{\partial x^4} + \frac{\partial^4}{\partial y^4} \right) D = 0. \quad (4.6)
$$

In the discrete domain this becomes:

$$
12D(i, j) + \\
D(i, j + 2) - 4D(i, j + 1) - 4D(i, j - 1) + D(i, j - 2) + \\
D(i + 2, j) - 4D(i + 1, j) - 4D(i - 1, j) + D(i - 2, j) = 0. \quad (4.7)
$$
We generate $A_s$, a smoothed version of $A$, which is filled with the new coefficients of Eq. (4.7). To preserve depth discontinuities on object boundaries, the boundary cuts are applied to the smoothness constraints as well. We then concatenate $A$ with $A_s$ and solve

$$
\begin{bmatrix}
A \\
\beta \cdot A_s
\end{bmatrix}
\begin{bmatrix}
x \\
b
\end{bmatrix} =
\begin{bmatrix}
b \\
0
\end{bmatrix},
$$

(4.8)

instead of the original $Ax = b$. The amount of smoothness required is configured by the parameter $\beta$. A high $\beta$ can cause over-smoothness while a low $\beta$ can generate weak smoothness. In all our experiments, we set $\beta = 0.01$. Note that the effect of smoothness and gradient refinement are very different. The latter is designed for removing sharp artifacts while maintaining the rest of the image intact; smoothness, however, is for adding a delicate touch to all depth textures. If smoothness is used for removing sharp artifacts it may cause over-smoothing, while applying a strong gradient refinement can damage essential gradients.

**Creating Final Output:** To form the final converted 2D+Depth output, we normalize the estimated depth ($x$ in Eq. (4.8)) between $(0, 255)$ and combine it with the query image. The final estimated depth for 4.3(a) is shown in Fig. 4.3(h), which includes all steps with smoothness. Our method produces a smooth depth that correctly resembles the depth of the players, field and audience. Furthermore, the ‘card-board effect’, where a constant depth is assigned to each player, does not occur in our method. We show this by zooming on a player depth block in Fig. 4.3(h) and amplifying it by normalizing the depth values of the block to the range of $(0, 255)$. The zoomed and amplified version of the yellow marked block in 4.3(h) is shown in Fig. 4.3(i). The player in the marked block demonstrates the strength of our gradient-based approach in estimating small depth details. It can be seen that different body parts of the player have different depth values.

### 4.4.4 Spatio-temporal Poisson Reconstruction

While the Poisson reconstruction technique discussed in Sec. 4.4.3 produces plausible results, one of its main limitations is maintaining temporal smoothness. Since the depth estimation is performed independently for each frame, the generated depth maps are not temporally smooth and can vary significantly between consecutive frames causing a flickering effect. While this limitation can be partially handled by temporally smoothing the depth maps during a post-processing phase, it is much more effective if we eliminate the problem from the source, and enforce temporal smoothness during the core depth estimation process. In order to do so, we enhance the Poisson reconstruction formulation such that it utilizes temporal gradients in addition to spatial gradients when reconstructing the depth. That is, instead of computing the depth of each frame independently, the information from the next and previous frames is also considered.

One of the main challenges, however, is utilizing this temporal information in the depth estimation process without limiting its parallelizable feature. Being parallelizable is an important aspect of our method, which enables processing different frames in parallel due to their independence. Con-
sidering temporal information, however, introduces dependence among frames. Therefore, in order to maintain the parallelizable feature, we determine a window around each frame and process each window independently. Within each window the depth maps of all frames are generated together and coherently. The final depth map for each frame is the average of all depth maps generated for that frame in different windows. While a bigger window size can achieve an overall better temporal coherence, it will significantly increase the computational complexity and decrease efficiency. Our experiments in Sec. 4.6.3 show that a window size of 3 (one frame before and one after) yields good results, and not much gain can be achieved by further increasing the window size.

For each window, we perform block-based matching, depth gradient mapping and refinement for all frames. We then enforce temporal smoothness by modifying Eq. (4.3) as in Eq. (4.9) for each of the frames within the window. In Eq. (4.9), $D_{\text{next}}$ and $D_{\text{pre}}$ refer to the next and previous frames respectively, and $(i_c, j_c)$ refers to the corresponding pixel in the neighbouring frame. In order to identify the corresponding pixels between each two consecutive frames, we use optical flow [81], which computes the horizontal and vertical displacements for all pixels. For the first and last frames in the window for which one of the neighbours does not exist, the non-existing connection will be removed.

$$D(i, j + 1) + D(i, j-1) - 6D(i, j) +$$
$$D_{\text{next}}(i_c, j_c) + D_{\text{pre}}(i_c, j_c) +$$
$$D(i+1, j) + D(i - 1, j) = \nabla \cdot G(i, j).$$  (4.9)

Temporal smoothness implies that the depth value of each pixel and its corresponding pixels in the next and previous frames should be similar. In other words, temporal smoothness implies that the temporal gradient should be set to zero. As a result, while the left hand side of Eq. (4.9) is an extension of Eq. (4.3) which includes temporal neighbours in addition to the spatial ones, $\nabla \cdot G(i, j)$ is still calculated using Eq. (4.4) which includes only the spatial gradients.

When formulating a solution in the form of $Ax = b$, we generate the matrix $A$ according to Eq. (4.9) such that it contains all frames in the window. Thus the size of $A$ will be $HWN \times HWN$, where $N$ is the window size. Finally, we concatenate $b$ and $x$ for all frames in the window, and solve for the depth maps ($x$). The estimated depth maps are then normalized between $(0, 255)$ collectively.

Note that since neither the optical flow nor the object masks are perfect, there is a chance that a pixel marked as an object (according to the mask) is recognized as a corresponding pixel to a non-object pixel in the neighbouring frame or vice versa. Establishing such temporal connections can cause fading of the object boundaries, as the two sides of the boundary will be connected through a temporal route. In order to solve this problem we first make sure that each two corresponding pixels have the same mask value before establishing a connection between them. Otherwise, we would remove the temporal connection by setting the corresponding connection in $A$ to 0.
4.5 Object Mask Creation

In order to have clear depth discontinuities on player boundaries, we delineate object boundaries. If object boundaries are not specified, the depth of players will blend with the ground, causing degradation in the depth quality. To detect objects boundaries we first create object masks. These masks are created automatically in a pre-processing step where motion and appearance are used to detect objects. While object segmentation for videos with simple motion or static scenes can be performed using methods such as [68], it is rather challenging for videos with complex motion. Therefore, we propose two different methods for object detection: one for close-ups, and another for non close-ups. Close-ups are characterized by small playing areas and large player sizes, while non close-ups usually have a larger field of view. As a result, a shot classification step is required prior to object detection. Shot classification takes an input image sequence, finds the shot transitions and classifies each shot as either close-up or non close-up. Based on the type of shot the appropriate object detection method is then applied. The method for non close-ups is mainly based on global features such as the color of playing field, while for close-ups, local features such as feature point trajectories [87] are used. In this section we discuss each step in details.

4.5.1 Shot Classification

Our shot classification stage has two main components: shot transition detection and shot classification. In shot transition detection, an input sequence is segmented into different shots by detecting the shot transitions. While there are several sophisticated techniques for handling shot transition detection [76], we designed a simple shot transition detection step suitable for our 2D-to-3D conversion method. Our implementation is designed to detect temporal impulsive changes in the frame structure. We predict each frame from its next frame using optical flow [81] and then estimate the global structure similarity between the original and predicted frame using SSIM [124]. A frame is flagged as a shot transition if: the global SSIM value is smaller than a certain threshold (0.7 in our experiments), and it increases in the next frame by at least 0.1. In other words, if the similarity between the predicted and original frames is low but it increases considerably as we move to the next frame, then there is a high chance that there is a shot transition.

The second step is to differentiate between two types of shots: 1) close-ups and 2) non close-ups. A close-up is defined as a shot with a small field area and large players. We use a color-based approach to detect the field area. We train a Gaussian Mixture Model (GMM) on samples collected from the playing fields and the white lines. In the test phase we estimate the log-likelihood of each pixel being generated by the learned GMM model. If the log likelihood is more than a threshold (−15 in our experiments), it is flagged as field area. The second discriminative cue for close-ups is player size. We exploit the observation that in close-ups, players often have a large size and the audience/ad banners are usually behind the player upper-body. Hence to measure the players size, first we invert the detected field so that white pixels indicate the non-field area. Then for each

---

\(^1\)This section was developed in collaboration with Mohamed Elgharib.
**4.5.2 Object Detection for Non Close-up Shots**

Object masks for non close-up shots are a fusion of background subtraction and non-field areas. The latter is estimated during the shot classification step. However, relying only on field detection to detect players can have a high missing rate. This is often the case for players of similar color to the field. Hence to generate a more complete detection, we fuse the field detection results with that of background subtraction. Background subtraction is a well known technique in video processing [71]. In this technique, first a homography is generated by warping all frames with respect to a reference frame (in our case it is the first frame). We use the method by Odobez et al. [88] with affine motion modelling to build our homography. The stationary background is then detected using temporal median filtering. Frame differencing between each frame and the stationary background is used to find the moving objects, which in our case are the players.

In order to further reduce player segmentation errors, we correct for possible misalignments between the frames and the stationary background through optical flow [81]. We perform frame
differencing using the local SSIM values for each pixel and the motion computed by optical flow. A pixel is flagged as a moving object if: 1) the similarity between that pixel in the frame and the stationary background is low (SSIM less than 0.4 in our experiments), and 2) the motion divergence in that pixel is high (higher than 0.01 in our experiments). Motion divergence measures the rate of spatial changes in motion [37]. Hence it is low in regions with high similarity (such as the field) and high otherwise (for players).

The final object mask is generated by a logical OR between the field detection based and background subtraction based approaches. Fig. 4.7 shows an example of a non close-up object mask creation. Note that some players are small from the field only method, but are fully detected using the background subtraction approach (green boxes). The opposite is true as well (yellow box). The fusion of both approaches nevertheless brings the best of both worlds with all players being fully detected.

4.5.3 Object Detection for Close-up Shots

In order to detect players in close-up shots we use a combination of frame-to-frame motion and feature point trajectories to obtain foreground and background matting strokes. Matting is then performed using these strokes and the generated mattes can be used as object masks after thresholding. However, in order to achieve cleaner results, we use field detection to remove possible mis-classified field areas.

Frame-to-frame motion is estimated through the optical flow method in [81], which provides us with a color coded flow field. We fit a GMM to the color coded flow field, and take the cluster with the most dominate Gaussian distribution as the camera motion segment. All other clusters are considered as the non-camera motion segment. This segmentation often has poor object boundaries and is not temporally coherent (see Fig. 4.8, approx. non-camera motion segmentation). Hence it
can not be used directly as object masks. Instead we combine it with sparse trajectories segmentation to obtain foreground and background matting strokes.

Sparse trajectories segmentation is obtained through extracting feature point trajectories and segmenting them into different groups [87]. This generates a sparse labelling for different objects (see Fig. 4.8, sparse trajectory segmentation).

In order to combine sparse trajectories segmentation with non-camera motion segmentation we estimate the overlap of each trajectory segment with the non-camera motion segment. If there is at least 30% overlapping, we label the trajectory segment as foreground (see Fig. 4.8, foreground), else background (see Fig. 4.8, background).

The feature point trajectories become the matting strokes and the method by Levin et al. [78] is used to extract a soft-mask of the players (see Fig. 4.8, matting). We then correct possible field mis-

Figure 4.8: For a close-up shot, a combination of feature point trajectories [87] and frame-to-frame motion [81] is used to generate background and foreground matting strokes. The method in [78] is then used to extract a dense players matte. Finally, field segmentation removes matting inaccuracies.
classifications by using the field detection of Sec. 4.5.1. This generates cleaner player boundaries. Finally, we threshold the generated mattes by 0.3 and get the final object masks (see Fig. 4.8, object segmentation).

4.6 Evaluation

All components of our proposed method have been implemented and compared against the closest system in the literature [71], and the ground-truth where available. For our experiments, both real and synthetic sequences have been considered. Note that the few parameters in our method are experimentally tuned once for all sequences. Specifically, we set the number of candidate images $K$ to 10, the block size $n$ to 9, the gradient refinement parameter $\alpha$ to 60, and the smoothness parameter $\beta$ to 0.01.

4.6.1 Examined Methods

Our 2D-to-3D conversion technique, which we refer to as DGC (short for Depth Gradient-based Conversion), is compared against several techniques as described below.

Original 3D: The original 3D-shot video that has been captured by stereo cameras. Results are compared subjectively.

Ground-truth Depth: Ground-truth depth maps are only available for synthetic sequences. As described in Sec. 4.4.1, they can be extracted from FIFA13 using PIX [11].

DT: The state-of-the-art method for data-driven 2D-to-3D conversion, Depth Transfer [71], trained on its own MSR-V3D database. MSR-V3D is available online and contains videos that have been captured using Microsoft Kinect.

DT+: Depth Transfer trained on our synthetic S-RGBD database. As stated in [71], capturing depth using Kinect is limited to indoor environments. This in addition to its erroneous measurements and poor resolution, limits Kinect’s ability in generating a large soccer database. In order to have a rigorous comparison, we trained Depth Transfer on our soccer database and compared it against our technique.

Depth from Stereo: For an objective comparison of our method against the original side-by-side 3D, we need to approximate the ground-truth depth. We do so using the stereo correspondence technique in [24]. While stereo correspondence techniques do not always produce accurate results, they can sometimes capture the overall depth structure and thus be used for objective analysis.

4.6.2 Subjective Experiments

To assess the visual 3D perception we perform several subjective experiments, and compare our method against the original 3D and DT+. We then demonstrate the benefits of our spatio-temporal Poisson reconstruction, specially for more temporally challenging scenes.
Setup

Our subjective experiments are conducted according to the ITU BT.2021 recommendations [16]. This recommendation suggests three primary perceptual dimensions for 3D video assessment: 1) picture quality, which is mainly affected by transmission and/or encoding, 2) depth quality, which measures the amount of perceived depth, and 3) visual (dis)comfort, which measures any form of physiological unpleasantness due to 3D perception, e.g., headache, eye strain, and fatigue. In our experiments, we measure depth quality and visual comfort. We do not examine picture quality as we do not degrade it using compression or transmission.

The test sequences were displayed on a 55” Philips TV-set with passive polarized glasses. The lighting conditions were low. According to the ITU recommendations, we set the duration of each sequence to be between $10 - 15$ seconds, and the viewing distance to be around $3 \text{ m}$ for videos with a resolution of $1280 \times 720$ and around $2 \text{ m}$ for $1920 \times 1080$. We used static and dynamic random dot stereograms to test their stereoscopic vision prior to the experiment. A stabilization phase was also performed before the actual experiments, where the subjects were asked to rate 4 representative sequences with 3D qualities ranging from best to worst. While these representative sequences were not part of the actual test, this phase was useful in stabilizing the subjects expectations and making them familiar with the rating protocol. The subjects were asked to ensure their full understanding of the experimental procedure prior to the actual test.

Evaluation of our Technique

For evaluating our 2D-to-3D conversion method, we show the subjects our converted sequences and measure their average satisfaction. We assess depth quality and visual comfort for four real soccer sequences using the single-stimulus (SS) method of the ITU recommendations. We carefully created the four soccer sequences using clips from original 3D videos such that each includes a different category of shots: long shots, medium shots, close-ups, and bird’s eye view. A long shot shows almost the entire field from a high camera position (Fig. 4.9, bottom right-most). In medium shots the camera is placed at a lower height and a smaller part of the field is visible (Fig. 4.9, top right-most). Close-ups have the camera zoomed on one or few players (Fig. 4.9, top left-most). In a bird’s eye view the camera is placed above the field (Fig. 4.9, bottom left-most). Fifteen subjects participated in this study, which are all computer science students and researchers. The sequences are shown to subjects in random order. Before displaying each sequence, a $5 \text{ sec}$ mid-grey field is displayed which indicates the coded name of the sequence. The $10 - 15 \text{ sec}$ sequence is then displayed, followed by a $10 \text{ sec}$ mid-grey field which asks the subjects to vote. The standard ITU continuous scale is used for rating. For depth quality, the labels marked on the continuous scale are Excellent, Good, Fair, Poor, and Bad, while for visual comfort the labels are Very Comfortable, Comfortable, Mildly Uncomfortable, Uncomfortable, and Extremely Uncomfortable. We asked the subjects to mark their scores on these continuous scales. Their marks were then mapped to integer values between 0-100 and the mean opinion score (MOS) was calculated.
Figure 4.9: Depth estimation for different types of shots using our method. Our method handles a wide variety of shots including Close-ups (e.g., top, left-most), Medium Shots (e.g., top, right-most), Bird’s Eye View (e.g., bottom, left-most) and Long Shots (e.g., bottom, right-most).

The MOS for all four sequences is shown in Fig. 4.10. For all sequences, DGC was rated in the range of Excellent by most subjects. For all figures in this chapter, error bars represent the standard deviation with a confidence interval of 68%. Examples of estimated depth maps are shown in Fig. 4.9. Note how DGC can handle a wide spectrum of video shots, including different camera views and clutter.

In addition, in order to show the potential of our method on field sports other than soccer, we examined four real non-soccer sequences containing clips from Baseball, Tennis, Field Hockey and American Football. Fig. 4.11 shows the MOS for these sequences. Field Hockey scored the highest as it resembles soccer the most. American Football scored the lowest, however. While some subjects reported very good depth, others reported the difficulty of depth perception due to the high dynamic environment of American Football with strong occlusions and clutter. Those subjects also reported a Mild Discomfort for the same reasons. It is important to note that the results on non-soccer are only meant to show the potential of our method, as we actually used the soccer database to convert them. For a high quality conversion of such sequences a proper database should be designed.
Comparison against Original 3D

We compare our converted videos against videos that are originally shot using stereo cameras. For this experiment, the Double Stimulus Continuous Quality Scale (DSCQS) method of the ITU recommendations is used. According to DSCQS, in order to assess the differences between each pair of sequences (original 3D and our converted 3D) properly, each pair should be observed by subjects at least twice prior to voting. Fifteen subjects participated in this study as well. The sequences were shown to them in random order without them knowing which is the original one. We then asked the subjects to rate depth quality and visual comfort for both sequences using the standard ITU continuous scale. Their marks are then mapped to integer values between 0-100 and used for calculating
the Difference Opinion Score (= score for DGC - score for original 3D). Finally we calculate the mean of the difference opinion scores (DMOS).

A DMOS of zero implies that our converted 3D is judged the same as the original 3D, while a negative DMOS implies our 3D has a lower depth perception/visual comfort than the original 3D. The DMOS of the soccer sequences for both depth quality and comfort is shown in Fig. 4.12. It can be seen that our conversion achieves comparable quality to the original 3D. This is especially true for long shots which account for around 70% of a full soccer game [30]. It is interesting to note that for some subjects our conversion was more comfortable than the original 3D. They reported that the popping out effect in original 3D was sometimes causing them discomfort.

![Figure 4.12: Difference mean opinion score (DMOS) between our converted sequences and the original 3D. Zero implies that our converted sequence is the same as the original 3D.](image)

![Figure 4.13: Difference mean opinion score (DMOS) between our converted sequences and Depth Transfer DT+. Positive DMOS means that our technique is preferred over DT+.](image)
Figure 4.14: Depth estimation for different sequences using: DT, DT+ and our method DGC. DT generates erroneous estimates, DT+ generates noisy measurements and does not detect players. Our technique outperforms both approaches.

**Comparison against State-of-the-Art**

We compare our conversion technique against Depth Transfer (DT+) [71]. Similar to the previous experiments, the study is done with fifteen subjects, the DSCQS method is used, and the DMOS is calculated for both depth quality and comfort. For this experiment, we examined the close-up and medium shot sequences since they are the most challenging sequences for 2D-to-3D conversion due to their wide spectrum of camera angles, occlusion, clutter, and complex motion. Fig. 4.13 shows DMOS for the medium shot and close-up against DT+. DT+ is outperformed by our method with an average of 12 points in close-ups and 15 points in medium shots. In addition, our technique was rated higher or equal to DT+ by all 15 subjects and the differences reported are statistically significant (p-value < 0.05). Fig. 4.14 shows some examples of extracted depth maps for DT, DT+ and our DGC. Note that as it can be seen in Fig. 4.14, the original implementation of Depth Transfer (DT) is much worse than DT+. Furthermore, it can be seen from Fig. 4.14 and Fig. 4.16 that the depth from DT+ can be very noisy sometimes, which in long term can cause eye strain.

**Effect of Spatio-temporal Poisson Reconstruction**

As discussed in Section 4.4.4, estimating depth independently for each frame may result in significant difference between the depth of consecutive frames. While simple shots may not suffer much from this problem and have a temporally smooth depth without the need of any further temporal enhancements, shots with more complex and detailed texture, such as close-ups, may suffer from significant variations in the depth maps of successive frames. This may degrade the quality of depth perception and cause visual discomfort.

While temporally smoothing the depth maps during a post-processing phase works well for simple shots, it cannot completely overcome the problem for temporally complex shots. Our spatio-temporal Poisson reconstruction method, however, generates temporally and spatially smooth depth
maps by utilizing temporal gradients in addition to spatial gradients during the depth calculation process. Thus, it can handle all types of shots and generate a comfortable and temporally smooth depth for all cases.

To assess the performance of our spatio-temporal method, we created two 10 sec sequences. The first one is composed of various shots from the four soccer sequences used in the previous experiments, which are all rather simple to handle. We refer to this sequence as Temporally Simple. The second sequence is composed of various temporally complex soccer shots that are difficult to
handle. In the figures, we refer to this sequence as *Temporally Complex*. The shots included in this sequence were not included in the four sequences previously used.

We showed the subjects three versions of each sequence: 1) Without any temporal smoothness. 2) Temporal smoothness applied as a post-process on the depth maps generated by a regular (spatio) Poisson reconstruction. For this we use the temporal smoothness provided by Karsch et al. [71] as part of their stereo-warping technique. 3) Temporal smoothness integrated in the depth generation process using our proposed spatio-temporal Poisson reconstruction, without any further post-processing refinements. We then assess depth quality and visual comfort for all sequences using the single-stimulus (SS) method of the ITU recommendations.

Ten subjects participated in this study. We showed them the sequences in random order and they were asked to rate depth quality and visual comfort using the standard ITU continuous scale. Fig. 4.15(a) shows MOS for the three versions of the *Temporally Simple* sequence. It can be seen that while no temporal smoothness causes degradation in the comfort and thus the depth quality, it can be fully resolved by post-processing. As a result, there is very little difference between the results of our spatio-temporal reconstruction and that of post reconstruction smoothing. However, the benefits of our spatio-temporal reconstruction become more clear in the *Temporally Complex* sequence, where post-processing is unable to fully overcome the problem. Fig. 4.15(b) shows MOS for this sequence. It can be seen that our spatio-temporal reconstruction improves the comfort by an average of 14 points compared to post reconstruction smoothing, and enhances the quality from Good to Excellent. The differences reported in this figure are statistically significant (p-value < 0.05).

### 4.6.3 Objective Experiments

We perform objective experiments on both real and synthetic sequences to measure the quality of our depth maps and compare it against the state-of-the-art. We then analyse the effect of our spatio-temporal Poisson reconstruction on temporal smoothness, and the effect of database size on depth quality.

**Comparison against State-of-the-Art**

For an objective comparison against state-of-the-art, we choose two sequences: a synthetic sequence and a real sequence. For the synthetic sequence we extract 2D+Depth for around 120 frames in the same way that the database was created (Sec. 4.4.1). In Fig. 4.16 (top) a frame of the synthetic sequence is shown followed by its ground-truth depth and estimated depth when using different methods (DT, DT+ and our DGC). All demonstrated depth maps are normalized and in the range of \((0 − 255)\). Results from DT are largely erroneous since the data in MSR-V3D hardly resembles soccer. While being trained on our database makes the results from DT+ significantly better, most players are yet not detected. Our technique DGC, however, manages to detect the players and generate a smooth depth that best resembles ground-truth. The Mean Absolute Error (MAE) against
Figure 4.16: Top row: Frame 3 of a synthetic sequence. Bottom row: Frame 24 of a real sequence. We show the depth extracted using: Ground-truth/Stereo Correspondence [24], DT, DT+ and DGC. Our technique DGC best reassembles the Ground-truth/Stereo Correspondence in both sequences.

ground-truth for the whole synthetic sequence is shown in Fig. 4.17 (a). As shown it the figure, the MAE of our method is much less than both DT and DT+.

Due to the absence of ground-truth depth for real sequences, performing objective analysis on them is challenging. In [71], Kinect depth was used as ground-truth. However, Kinect is incapable of capturing depth information in outdoor environments. As a result, Kinect cannot be used for generating ground-truth estimates for soccer matches. Instead, given a 3D-shot soccer sequence, we use stereo correspondence [24] to approximate the ground-truth depth map. Fig. 4.16 (bottom) shows a frame from one of the most challenging soccer test sequences. Its extracted depth, though not perfect, captures the overall depth structure which can be used for inferring the quality of the converted depth maps. The estimated depth maps using DT, DT+ and our DGC are also shown in Fig. 4.16 (bottom). It can be seen that our technique (DGC) best resembles the ground-truth. This is also captured objectively over a range of 100 frames. Fig. 4.17 (b) shows the MAE of this sequence against the extracted ground-truth. As shown it the figure, our method produces much lower MAE than DT and DT+.

**Effect of Spatio-temporal Poisson Reconstruction**

In order to demonstrate the advantage of our spatio-temporal Poisson reconstruction, we use the same two real and synthetic sequences (shown in Fig. 4.16). For each sequence, we generate the depth maps using a temporal window size of: *one* (without temporal smoothness), *three* (one frame before and one after), and *five* (two frames before and two after). Fig. 4.18 (a) shows the average depth values for each frame of the synthetic sequence. It can be seen that without temporal smoothness the scene experiences sudden changes from frame to frame, but the changes become smoother as the window size increases. Also, without temporal smoothness the difference between the maxi-
Figure 4.17: An objective comparison between our DGC method and the closest method in the literature DT, and its extension DT+ on a (a) synthetic soccer sequence and (b) real soccer sequence. Our method generates much lower MAE than both DT and DT+ for both sequences.

The maximum and minimum average depth value is around 70, while with temporal smoothness it is reduced to around 15. Results for the real sequence also show that while without temporal smoothness the difference between the maximum and minimum average depth value is around 110, this value is reduced to around 50 when temporal smoothness is applied (Fig. 4.18 (b)).

Fig. 4.19 (a) shows the MAE between the depth of each frame in the synthetic sequence and its previous frame. Each pixel is compared to its corresponding pixel in the previous frame, where the corresponding pixels are identified using optical flow. It can be seen that the MAE is reduced from a maximum of 57 (without smoothness) to a maximum of 3 (window size of 5). Fig. 4.19 (b) shows the MAE between the depth of each frame in the real sequence and its previous frame. It can be seen that applying temporal smoothness significantly reduces the MAE. However, there is not much gain in increasing the window size from 3 to 5.

Effect of Database Size

To investigate the importance of our S-RGBD database size we examined six different database sizes: 1000, 2000, 4000, 8000, 13000 and 16000 images. For this experiment, a synthetic sequence with 120 frames was generated. This sequence includes a wide variety of shots that can occur in a soccer match. Results show that up to a size of 8,000 images, due to the absence of big enough data the performance fluctuates around an MAE of 30. Starting from 13,000 images there is a boost in performance which reduces MAE to around 20. However, the performance stabilizes around 16,000 images. Therefore, a database size of 16,500 images was used in our evaluation.
Figure 4.18: The average depth values for each frame of a (a) synthetic sequence and (b) real sequence when using different temporal window sizes. While without temporal smoothness the scene experiences sudden changes of depth, the depth changes are much smoother when temporal smoothness is applied.

Figure 4.19: Mean Absolute Error (MAE) between each two consecutive frames of a (a) synthetic sequence and (b) real sequence, when using different temporal window sizes.
4.6.4 Computational Complexity

We measure the running time for DGC and DT+ averaged over 545 close-up frames and 1,726 non close-up frames. The spatial resolution is 960 $\times$ 1080 pixels. DGC takes 3.53 min/frame for close-ups and 1.86 min/frame for non close-ups. The average processing time for DT+ is 15.2 min/frame, which is slower than our technique in both close-ups and non close-ups. DGC requires more time for close-ups due to the more expensive mask creation step. As non close-ups can account for up to 95% of a soccer game [30], we can benefit from the faster non close-up processing. Nevertheless, we cannot ignore close-ups as they often contain rich depth information. All numbers are reported from processing on a server with six processors Intel Xeon CPU E5-2650 0 @2.00 GHz, with 8 cores, with a total of 264 GB RAM and 86 GB Cache.

4.7 Summary

In this chapter, we presented a 2D-to-3D video conversion method for soccer videos. Unlike previous methods, the proposed method can handle the motion complexities and the wide variety of scenes present in soccer matches. Our method transfers depth gradients from a synthetic database and estimates depth through a spatio-temporal Poisson reconstruction. We implemented our method and used both real and synthetic sequences for evaluating it. Our subjective and objective results show the capability of our method in handling a wide spectrum of shots with different camera views, colors, motion complexities, occlusion, and clutter. Our created 3D videos were rated Excellent by most subjects. In addition, our method outperforms the state-of-the-art both subjectively and objectively, in all real and synthetic sequences.

This work contributes three key findings that can impact the area of 2D-to-3D video conversion, and potentially 3D video processing in general. First, domain-specific conversion can provide much better results than general methods. Second, transferring depth gradient on block basis not only produces smooth natural depth, but it also reduces the size of the required reference database. Third, synthetic databases created from computer-generated content can easily provide large, diverse, and accurate texture and depth references for various 3D video processing applications.
Chapter 5

VR Content Generation from Regular Camera Feeds for Sports

5.1 Introduction

Virtual reality is currently experiencing a growing interest in the multimedia industry. Despite large investments from giants such as Facebook, Google, Microsoft, Samsung, and others, one problem remains that prevents VR from being adopted on a wider scale; the lack of real content. The vast majority of current content is synthetic, generated for the gaming industry. The only approach for generating real content is by using VR capturing devices. Such devices, commonly referred to as VR camera rigs, contain multiple cameras stacked next to each other in a way that maximizes the field of view [4, 19, 13]. Camera outputs are then stitched together to enhance the overall sense of immersion.

Current solutions for capturing high-quality VR content require upgrading the entire production pipeline. This includes data capturing, processing and distribution. Such upgrade is expensive to set-up and operate, which makes it an unappealing solution. We propose an alternative approach for VR content generation that converts the traditional broadcast to VR through a post-processing stage.

Most large sporting stadiums contain multiple high-end cameras, capturing the field from different positions. These cameras are operated by a professional filming staff that can create production quality content. For instance, broadcasting a FIFA World Cup game often involves more than 20 cameras [47]. These cameras capture the field from different angles and different positions, including a few main cameras positioned on the halfway line, a high left and a high right camera. The report in [111] shows the most common camera positions for different field sports, including basketball, and ice hockey. In most games, there is at least one main camera positioned usually in the middle of the field. This camera follows the action with a wide angle covering around 50% of the field. With such setup already in place, we propose a solution for high-quality VR content
generation. Our solution utilizes existing video feeds in a post-processing step without the need of upgrading the entire production pipeline.

Producing VR content from the traditional video feeds is a quite complex task and requires addressing multiple challenges. First, we need to widen the field of view to at least 180 degrees. We achieve this by utilizing the motion of the main camera to generate a wide-angle static panorama. The field area is then overlaid on the panorama using Poisson blending to allow seamless blending. Second, due to the limited coverage of the main camera, players tend to appear/disappear throughout the recording. Such effect significantly degrades the feeling of immersion. To overcome this problem, we identify and retrieve the missing players from different camera feeds. For this, all feeds need to be aligned with respect to a reference frame, which is challenging because of the large distance between cameras. This distance causes the position of objects and the orientation of lines to appear differently when viewed from different cameras. This effect is referred to as parallax. Large amounts of parallax cause state-of-the-art image alignment techniques, e.g. [130], to fail. To address this problem, we remove parallax by first obtaining camera parameters, and estimating the 3D position of each pixel. We then warp each feed to the position of the main camera.

To evaluate our method, we captured sports games using 3 regular cameras and a 360 camera simultaneously. Using our technique, we generated VR content from the 3 regular video feeds, which were positioned on the left, middle, and right side of the field. We conducted subjective studies in which participants were asked to rate their sense of presence and perceived video quality when viewing our generated content. In addition they were asked to compare our content with content captured using the 360 camera. Our results show that most participants rated their sense of presence between Good to Excellent. Also, our generated content was rated similar to the captured 360, and did not introduce any major artifacts. Analysis on missing players show that our method retrieves missing players accurately with a maximum displacement error of around $20 cm$.

The rest of this chapter is organized as follows. Section 5.2 provides a summary of the related work. Section 5.3 describes our proposed method in details. Section 5.4 presents an extensive evaluation of our technique, and Section 5.5 concludes the chapter.

### 5.2 Related Work

Recently, VR has gained strong interest from both the industrial and research communities. Stengel et al. [112] proposed an affordable solution for VR headsets which incorporates eye tracking and tackles motion sickness. Perrin et al. [98] addressed quality assessment of VR content through a multi-modal dataset, while Chang et al. [34] proposed methodology for quantifying the performance of commodity VR systems. Zare et al. [131] presented a streaming solution for VR that can reduce the bitrate down by 40%. A survey on VR streaming solutions is presented in [42]. While this line of research addresses a wide range of VR topics, they do not address the content generation step.

Capturing VR content requires a camera rig with a field of view of 180 to 360 degrees. A number of such rigs have been recently introduced, including GoPro’s Omni [5], Samsung’s Beyond [13],
Facebook’s Surround 360 [4] and Google’s Odyssey [19]. These rigs contain multiple cameras stacked next to each other in a way that maximizes the field of view. Various software tools are used to allow seamless stitching and blending of the different camera views. Specialized filming teams are needed to operate the production pipeline. Companies offering such solutions include NextVR [8] and Jaunt [6]. However, using any of these solutions requires upgrading the entire production pipeline. This is often an expensive and inconvenient process.

VR content generation is based on creating a wide-view panorama of the scene of interest. One approach for panorama generation is through image stitching. Here, images are aligned in space by estimating a warping field through feature point matching. Many techniques assume simple camera rotations [23] and/or planar scenes [18]. Others relax these constraints through dual homography [51] or by smoothly varying the affine/homography [80, 130, 33]. Zaragoza et al. [130] rely on projective transformation and estimate local homography for alignment. Chang et al. [33] use a parametric warp which is a spatial combination of a projective transformation and a similarity transformation. Perazzi et al. [96] use patch based error metric similar to optical flow to estimate warping.

Most current stitching techniques allow only a small parallax and hence assume images are taken from nearby cameras. Recently, Zhang et al. [132] proposed an approach that relaxes this assumption. A mesh-based framework is proposed that optimizes for both alignment and 2D regularity. Interesting results are generated that can handle moderate parallax and moderate deviation from planar structures. However, limitations still exist such as the inability of handling straight lines across multiple cameras. Line straightness can only be preserved locally in each image, but not across images. Such artifacts are problematic for sports content, as it is often crucial to preserve the straightness of field lines.

Another approach for generating panoramic images is through 3D modeling and texture mapping. Multiview techniques perform 3D model reconstruction by estimating point locations via feature correspondence. VisualSFM [125] provides a GPU based optimized implementation of such techniques. Generating dense 3D reconstruction in outdoor environment is still a challenge. While the technique in [67] produces good results with large datasets, the reconstruction quality highly depends on good feature point correspondence. Such correspondence is not necessarily available in sports data due to the low textured nature of playing fields.

Sports content has special properties. Specifically, the presence of all players and the straightness of the lines are of major importance, while the background is less of a concern. The main component of sports VR content is a wide-view panorama with all players present at every time instant. Fehn et al. [45] use two nearby high resolution cameras to generate a high resolution (5k) panorama of a soccer match. The two cameras are planted in a way to have a full coverage of the field. This deployment, however, is a special setup which is difficult to achieve with the current broadcast systems.

Inputs from multiple cameras have been used to create free view point video (FVV) [20, 52]. In FVV, the task is synthesizing novel views from the available ones. This process, however, contains a
number of stages such as camera calibration, segmentation, depth estimation and 3D reconstruction. Since each step is error-prone, high quality FVV assumes strong constraints on the underlying cameras. Multiple works, e.g. [60, 85], proposed a number of FVV approaches for soccer. However, they all require pre-calibrated static cameras. Such set-up is hard to satisfy in sporting events with highly dynamic nature. Germann et al. [52] presented a FVV technique that robustly handles multi-cameras with wide baselines in an uncontrolled environment. Feature correspondence between cameras are found and back-projection errors are used to estimate a novel scene reconstruction method. Angehrn et al. [20] acknowledged that aligning multiple cameras is one of the most challenging tasks for FVV. To improve the performance of this step, they introduced the concept of a static high resolution master camera. All cameras are aligned to this camera, and a frame-to-frame recursive alignment is used.

A wide-view panorama can also be generated using different time instants of a single camera. This requires aligning each video frame to a reference panorama plane. Ghanem et al. [53] proposed an approach by matching global features such as image patches rather than matching small salient points. Their approach, however, does not consider the temporal stability of the estimated alignments and hence may generate shaky results. Carr et al. [32] proposed a gradient-based alignment to edge images. Due to computational complexity of the approach, the calibration does not scale well with video.

**Summary/Motivation:** Despite the rich literature of image stitching and panorama generation techniques, up to our knowledge there is no prior work on producing VR content from the traditional broadcast pipeline. We present a computational approach for doing so and we tailor our solution for sports. Our solution exploits the movement of a main camera to generate a wide-view video panorama and utilizes nearby cameras to estimate missing data such as players. We optimize the visual quality of our results using careful image blending and accurate alignment suitable for our problem.

### 5.3 Proposed Method

#### 5.3.1 Overview

The output of our proposed technique is a 360 video in equirectangular format that can be viewed on VR headsets or regular displays using 360 video players. In order to generate this 360 video from regular camera feeds, one of the cameras is chosen as the user’s viewing position. We refer to this camera as the main camera. While the best choice would be a wide-view camera that follows the action, any rotating camera can be sufficient for our method. Note that with multiple choices for the main camera, multiple VR videos can be generated from different positions and angles, providing the user with multiple options. For any chosen main camera, all nearby cameras within its 180 degrees field of view can be used as complementary feeds and help in covering the whole field.
Our technique consists of three stages (Fig. 5.1). In the first stage, we use the main video feed to generate a wide-view static panorama by means of image registration and median filtering. This panorama is used as the background of our 360 environment, in which the field and players will be then overlaid on. Although the background remains static, subjective studies show that it has little impact on the sense of presence, as it is not the focus of attention. In the second stage, we remove parallax between all video feeds by warping them to the position of the main camera. In the third stage, we use Poisson blending to first overlay the main feed on the panorama, and then copy the missing players from the complementary feeds. In the following sections, we describe each stage in more detail.

5.3.2 Generating Static Panorama

The viewing angle in regular sports videos is usually not wide enough for an immersive experience. In order to improve the sense of presence, a wider viewing angle is needed. As a result, we increase the viewing angle by utilizing the camera rotation, and generating a panorama image which includes the static parts of the scene. This stage can be performed only once, or periodically during a long game to capture any changes in the background. Only the main video feed is used in this stage. It is recommended to use a shot in which the camera rotates over a large angle and with minimum zoom. This widens the viewing angle of the generated panorama. In order to display a viewing angle greater than 180 degrees, we perform a planar to spherical conversion on each frame prior to
any processing. The camera rotation is then transformed to a wider viewing angle by aligning the spherical frames using image registration techniques, and applying median filtering. Fig. 5.2 shows an example static panorama generated from a basketball game.

**Conversion to Spherical:** Aligning planar images using projective transformation can cause shape and size distortion. This problem becomes more severe as the angle between the frames increases. While methods such as [33] try to overcome this problem, a viewing angle above 180 degrees cannot be achieved in planar format. In order to produce a panorama image with a viewing angle above 180 degrees from planar video frames, we convert the frames to spherical projection. The equirectangular projection is a standard way of projecting the 3D world onto a flattened sphere. To map a frame to an equirectangular projected image, for each pixel in the frame \((x, y)\), the spherical coordinates \((\theta, \phi)\) are calculated using Eq. (5.1). Here, the origin is taken at the centre of the frame. \(\alpha\) is the camera angle. \(Z_{img}\) is the distance of the frame from the camera, and has a constant value for a given camera which is a function of its focal length. The pixel is then mapped to \((r\phi, r\theta)\) in the equirectangular image. Note that \(r\) is constant, and can be chosen arbitrarily based on the desired output size (resolution).

\[
\varphi = \arctan\left(\frac{x}{Z_{img} \cos(\alpha) + y \sin(\alpha)}\right),
\]

\[
\theta = \frac{\pi}{2} + \arctan\left(\frac{Z_{img} \sin(\alpha) - y \cos(\alpha)}{\sqrt{(Z_{img} \cos(\alpha) + y \sin(\alpha))^2 + x^2}}\right).
\]  

**Image Registration:** By performing image registration on the equirectangular frames, we transform the camera rotation to a wider angle of view. We automatically perform registration using feature based image registration techniques such as [23]. First, we extract and match SIFT features [84] between consecutive frames. Using RANSAC (random sample consensus) [48] we select a set of inliners that are compatible with a homography between the frames. We then align the frames according to the homography by applying a similarity transformation.

**Median Filtering:** The static panorama is extracted from the aligned frames using median filtering. We assign the value of each pixel in the panorama to be the median across all aligned frames. By applying median filtering, the moving objects will be removed, leaving only the static
Figure 5.3: Sample frames of 3 different video feeds positioned around the field. All frames are shot at the same time instant.

Figure 5.4: Removing parallax between the right camera frame in Fig.5.3(c) and the frame from the main camera in Fig.5.3(b) using its object mask. The object mask is generated using the background subtraction technique.

background. Note that applying median filter only on key frames can generate sharper results than applying it to all frames.

5.3.3 Removing Parallax

In a regular sports production the cameras are usually placed meters away from each other, causing a huge amount of parallax between them. Fig. 5.3 shows an example of 3 frames from cameras in different positions captured at the same time instant. Notice how the parallax affects line orientations and player positionings. In this section, we describe how we warp the camera feeds to remove such parallax.

Obtaining Relative Camera Parameters: In order to remove the parallax, we warp all feeds to the position of the main camera. We do so using relative camera positions and the 3D position of each pixel in space. While the availability of such information is desirable and can further improve the results of our technique, it is more practical not to always assume such information is given. Thus, we obtain an estimation of the relative camera parameters using the VisualSFM [125] software, and estimate the 3D position of each pixel using plane fitting and object masks. VisualSFM performs a sparse 3D model reconstruction using structure from motion techniques and provides an estimation of the relative camera positions (C), relative camera rotation matrixes (R), camera focal lengths (f), and sample 3D points (X^w_w, Y^w_w, Z^w_w).

Generating Object Masks: Object masks are used throughout our technique in different stages and for multiple purposes. They are required for estimating the 3D position of the players, identi-
fying the missing players, and copying them on the panorama. An object mask indicates the pixels which are not part of the static background. For sports, this is mainly the players. To create these masks we use the background subtraction technique. We first apply a median filter on every group of frames to get the background. Choosing a larger group size can better identify slower moving objects, such as a player that is still for a few seconds, but it may introduce more noise. The moving objects are then detected by subtracting each frame from its background. We further enhance the object masks by applying several morphological filters. Fig. 5.4(a) shows the generated object mask for the frame shown in Fig. 5.3(c).

**Estimating the 3D Position of Pixels:** When capturing an image, a 3D point \((X_w, Y_w, Z_w)\) in world coordinates is first projected to the camera coordinates \((X_c, Y_c, Z_c)\) through Eq. (5.2). In this equation, \(R\) is the rotation matrix of the camera, and \(T\) is the translation vector calculated based on the camera positions \((T = -RC)\). The 3D point is then projected on the 2D image through Eq. (5.3), where \((x_i, y_i)\) represent the image coordinates with the origin being at the image centre. Due to the loss of the third dimension in Eq. (5.3), this projection is non-reversible unless \(Z_c\) is known. As a result, to find the 3D position of a pixel, we should estimate its \(Z_c\).

\[
\begin{bmatrix}
X_c \\
Y_c \\
Z_c
\end{bmatrix} = [R|T] \begin{bmatrix} X_w \\
Y_w \\
Z_w \end{bmatrix} \tag{5.2}
\]

\[
\begin{bmatrix}
x_i \\
y_i
\end{bmatrix} = \begin{bmatrix} fx_c/Z_c \\
fy_c/Z_c
\end{bmatrix} \tag{5.3}
\]

In the camera’s coordinate system, the field is a plane parallel to the \(x\) axis which can be presented as \(bY_c + cZ_c = 1\). Since usually a large area of the frame is covered by the field, we estimate the plane parameters \(b\) and \(c\) by fitting a plane to sample \((X_c, Y_c, Z_c)\) points. To obtain such samples, we use Eq. (5.2) to project the sample 3D points provided by VisualSFM to the camera coordinates. With the plane parameters in hand, \(Z_c\) of each field pixel \((x_i, y_i)\) can be estimated through Eq. (5.4). From the non-field areas, our main concern is the players, which are indicated by the object mask. Based on the object mask, we estimate the \(Z_c\) of each player pixel to be the \(Z_c\) of the place where the player’s feet touch the ground.

\[
Z_c = \frac{1}{by_i + c}. \tag{5.4}
\]

**Warping:** Using the 3D pixels positions and the relative camera parameters, we warp each video feed and its corresponding object mask to the position of the main camera. To do so, for each pixel \((x_i, y_i)\), we first revert the camera projection to find the world coordinates. We then project each point back to a 2D image, where the new coordinates \((x_{i\text{main}}, y_{i\text{main}})\) are calculated according to the main camera parameters. This process is shown in Eq. (5.5) and Eq. (5.6). In
Eq. (5.5), an accurate estimation of the relative camera positions $\mathbf{C}$, which manifest itself in the translation vectors $\mathbf{T}$, can successfully remove all parallax. Note that, as a result of such warping, parts of the field that were originally occluded by the players may now become visible, causing empty shadow-like holes under each player. Such holes can be filled by inpainting techniques. In our experiments, we use an averaging approach for filling such holes. Fig. 5.4(b) shows an example of our warp applied to Fig. 5.3(c). Notice the similarity between the warped image and the main feed (Fig. 5.3(b)) in the field lines orientations and positioning of the players.

$$
\begin{bmatrix}
X_{c_{\text{main}}} \\
Y_{c_{\text{main}}} \\
Z_{c_{\text{main}}}
\end{bmatrix} = R_{\text{main}}(R^{-1}(\frac{Z_{c}}{f})
\begin{bmatrix}
x_i \\
y_i \\
f
\end{bmatrix} - T) + T_{\text{main}}
$$

(5.5)

$$
\begin{bmatrix}
x_{i_{\text{main}}} \\
y_{i_{\text{main}}}
\end{bmatrix} =
\begin{bmatrix}
f_{\text{main}}X_{c_{\text{main}}}/Z_{c_{\text{main}}}
\\
f_{\text{main}}Y_{c_{\text{main}}}/Z_{c_{\text{main}}}
\end{bmatrix}
$$

(5.6)

### 5.3.4 Overlaying Using Poisson Blending

In order to generate our output 360 video, we need to overlay parts of each video feed on the static panorama. To seamlessly blend the copied parts with the background, we use Poisson image editing. Poisson image editing is known as a seamless image cloning algorithm based on gradient field [97] and it produces more plausible results than just simply overlaying the objects on the panorama. However, a limitation of the Poisson image editing approach is that the color of the source image gets totally adapted to the background image. To overcome this problem, we utilize an image cloning algorithm based on a modified Poisson problem [116]. The modified version has a color preserving parameter. A large color preserving parameter perfectly preserves the color of the source and background in the overlaid result.

For each frame, we first overlay the main feed. Players missing from the main feed are then identified and copied from the complementary feeds. A main feed that follows the action is most likely to cover the ball and most players, leaving only a few players missing.

**Overlaying Frames from the Main Feed:** To overlay a frame from the main feed on the panorama, we first convert it to spherical format using Eq. (5.1). We then perform image registration by matching SIFT feature points between the frame and the panorama, and using RANSAC to select a set of inliers. After aligning the frame with the panorama by applying a similarity transformation, we seamlessly blend the frame borders by means of Poisson blending. Note that in order to reduce the effect of possible misalignments, if a player is on the borders of the frame, it is removed and considered as a missing player. Fig. 5.5(a) shows a sample frame from the main feed (Fig. 5.3(b)) overlaid on the static panorama.

**Overlaying Missing Players:** Missing players are identified using our object masks. If an object in the mask is partially or completely outside the area covered by the main feed, it is considered missing and is copied. For example, after warping the right view object mask in Fig. 5.4(a) to the
Figure 5.5: Overlaying the main feed on the static panorama, and copying the missing players from the left and right feeds. The blue arrows indicate that the players are copied from the left or right feeds.

position of the main camera (similar to Fig. 5.4(b)), 2 of its objects fall into the area covered by the main feed (Fig. 5.5(a)). The other 3 objects, however, are identified as missing and are copied.

Similar to the main feed, for identifying and copying the missing players, we should first align the warped complementary frames with the panorama. With the parallax removed, this alignment can be performed rather accurately. However, for a better alignment, and to overcome possible errors in estimating the camera parameters (Sec. 5.3.3), we perform image registration on planar images. To do so, we keep the warped frames in planar format and convert the field area of the panorama from spherical to planar format using Eq. (5.7). We then use SIFT and RANSAC to calculate the homography, and align images by applying a projective transformation. Missing players are then identified, copied, and blended seamlessly with the panorama using Poisson blending.

Finally, after all missing players have been overlaid, the planar panorama is converted back to spherical projection and placed in its corresponding location in the 360 panorama. Fig. 5.5(b) shows a zoomed in version of a final 360 frame after all missing players (blue arrows) have been overlaid.

\[
x = \tan(\varphi)(Z_{img} \cos(\alpha) + y \sin(\alpha)),
\]

\[
y = Z_{img} \frac{\sin(\alpha) - \tan(\theta - \frac{\pi}{2}) \cos(\alpha)}{\cos(\alpha) + \frac{\tan(\theta - \frac{\pi}{2}) \sin(\alpha)}{\cos(\varphi)}},
\]

(5.7)
Figure 5.6: Examples of final panoramas generated by our technique for different games: basketball (top), hockey (middle), and volleyball (bottom). The blue arrows indicate the players that have been copied from the left or right feeds.

5.4 Evaluation

To evaluate our VR content generation technique, we conduct subjective studies to measure the average subject satisfaction when observing our generated content. We also compare our results against content captured using 360 camera. In addition, we analyze the accuracy of our technique in retrieving missing players.

Our technique requires sports video feeds from different cameras around the field. At least one of the cameras needs to be moving. While such setup is realistic for broadcasting companies [111], we do not have access to their captured feeds. In addition, to the best of our knowledge, all available datasets such as [50, 114, 99] only provide feeds from static cameras. Hence, we captured our own data, while simulating broadcasting setups.

5.4.1 Setup

We captured multiple games using a GoPro Omni 360 camera as well as 3 individual GoPro Hero4-black cameras. All cameras captured the scene simultaneously. The Omni camera rig consists of 6 GoPro Hero4-black cameras, capturing in different directions. It was deployed in the middle of the field to capture 360 content. We treat this captured 360 content as ground-truth and compare it against our own reconstruction. The 3 individual cameras were deployed at the left, right and middle of the field, capturing the scene in 4K resolution. We synced the 3 individual cameras by
pairing them with the GoPro Wi-Fi remote. The synchronization was further refined manually. The middle camera rotates with the action, and is considered as the main feed. The left and right cameras are static. Initially, the middle camera is rotated with a wide angle so it would capture most of its surrounding and cover around a 360 degrees. GoPro Hero4-black cameras are wide-angled and do not provide zooming options. Hence, to simulate professional content more accurately, we zoom on our 3 individual camera feeds in a post-processing step. For this we use the GoPro Studio software.

While our technique can be used for all field sports, we used data from 3 different games: basketball, ice hockey, and volleyball (Fig. 5.6). From each game, we chose a 30-second sequence and converted it to VR content using our technique. For the same sequence, we also stitched the GoPro Omni feeds using its recommended software (Autopano Video) to create what we refer to as the original 360 content.

In our subjective experiments we assess the sense of presence and the video quality for both the original 360 and our generated content. A high sense of presence means that the participants are fully immersed into the action. For video quality, we focus on the amount of artifacts. Generating VR content relies heavily on image processing techniques and is therefore prone to various artifacts. By assessing the quality we measure the amount and visibility of such artifacts.

5.4.2 Evaluation of our Technique

We conduct a subjective study to measure the average subject satisfaction when viewing our generated content. Fifteen participants took part in our experiments. They were all computer science students and researchers. We used Oculus Rift to display the VR content. We displayed the games in random order. Prior to the actual experiments, we showed the participants samples of professionally produced 360 videos from the Rio olympics. This familiarized the participants with the VR device and the 360 environment and hence stabilized their expectations. We noticed that participants tend to move their head more when they first wear the device, and focus more on the games as they get used to the experience.

We used the standard ITU continuous scale to rate both video quality and sense of presence. The labels marked on the continuous scale are Excellent, Good, Fair, Poor, and Bad. We asked participants to mark their scores on the continuous scales. Their marks were then mapped to integer values between 0-100, and averaged to calculate the mean opinion score (MOS). Participants were asked to clarify all their questions and ensure their full understanding of the experimental procedure.

Fig. 5.7 shows the MOS for different games. For all figures in this chapter, error bars represent the standard deviation with a confidence interval of 68%. Most participants rated both video quality and sense of presence in the range of Good to Excellent for all games. This means that they were well immersed in the 360 experience. Between the three games, hockey has the least score. This is because the low textured hockey field makes it difficult to perform accurate feature matching and alignment.

In addition, after the experiment, we asked the participants whether they noticed that the background was static. While some participants didn’t notice it at all, as they were focused on the field
and players, the ones that did, stated that it had affected their sense of presence marginally. Note that in our technique the background can change periodically at the expense of more computational cost.

5.4.3 Comparison against Original 360 Content

We compare the results of our technique against original 360 content captured at the same time instant using a 360 camera. For this experiment, we use the double stimulus method (DSCQS), where participants view both content in random order and can re-view them as many times as they need. Participants were asked to rate the video quality and sense of presence for both content using the standard ITU continuous scale. Their marks were then mapped to integer values between 0-100. We calculated the mean of difference opinion scores (DMOS) by averaging the difference opinion scores (= score for our technique - score for original 360). A DMOS of zero implies that the results of our technique are judged the same as the original 360, while a negative DMOS implies that our result has a lower quality/sense of presence than the original 360.

Fig. 5.8 shows the DMOS of both video quality and sense of presence for each game. The small DMOS values indicate that most participants found their immersive experience to be quite the same when comparing our generated content against the ground-truth content captured using 360 camera. In addition, the only statistically significant difference reported (p-value < 0.05) is the sense of presence for hockey.

5.4.4 Analysis of Player Misplacements

Copying the missing players is an important aspect of our technique. Failing to accurately place the players at their correct location can cause sudden player movements that may seem unnatural and disturbing. Fig. 5.6 shows examples of final panoramas generated by our technique. Note that in

Figure 5.7: Mean opinion scores of video quality and sense of presence for different games.
these examples some feeds were zoomed more than usual, in order to have more missing players for demonstration purposes. The blue arrows indicate the missing players that were copied from the left and right feeds. To analyze the effectiveness of our technique in retrieving these missing players, we measure the amount of their displacement. We use the originally captured wide-angle main feed as reference. We measure the distance between the position of each copied player and its original position in the reference frame. We define the position of a player, as the pixel coordinates of the place where the player’s feet touch the ground. Fig. 5.9 shows the average displacement. Error bars represent the standard deviation. It can be seen that the displacement is highest for hockey, with a maximum around 10 pixels. This is because hockey is more prone to misalignment errors due to its high intensity field color with low textured regions. However, we should note that a displacement of 10 pixels in the panorama translates to a distance around 20 cm in a real field, which is fairly small.

5.5 Summary

We presented a technique for generating VR content for sports from common broadcast camera feeds. While current solutions for producing high-quality VR content require upgrading the entire production pipeline, our technique utilizes the existing camera setup to generate immersive content. We assume the presence of at least one camera with rotational movement and two or more complementary cameras which altogether cover the whole field. Note that for more visually pleasing results the cameras should be sufficiently far from the field. This is because our technique is prone to placing the players incorrectly oriented in space. For example, it might appear like they are moving sideways. This effect will increase as the complementary cameras get closer to the players or
more distant from the main camera. To solve this issue a complete 3D reconstruction of the missing players is needed. This, however, will require having many cameras capturing these players from different directions. Our data generation method has three main stages: (1) creating a wide-angle panorama, (2) removing parallax and aligning all video feeds, and (3) overlaying the field and the missing players on the panorama by means of Poisson blending. Subjective experiments show that our results are comparable with the original 360 content in terms of both quality and sense of presence. In addition, MOS ratings indicate that most participants experienced a strong sense of immersion. Objective analysis shows that the player misplacement error of our technique is small (less than 10 pixels).
Chapter 6

Conclusions and Future Work

6.1 Conclusions

Due to the availability of immersive displays and devices in the market, immersive videos are becoming increasingly popular. However, generating and streaming immersive content involves many challenges. In this thesis, we focused on the challenges of transmission, customisation, and conversion of immersive videos. Specifically, we proposed (1) new system for streaming 3D videos to heterogeneous devices, (2) data-driven 2D-to-3D conversion technique, and (3) technique for generating VR content from regular broadcast feeds.

Because of the high variety of 3D displays in the market, depth customization becomes very important; otherwise the content can appear rather flat or cause discomfort. Also, different display technologies should be supported, which each require a different type of input. Thus, streaming and rendering 3D stereoscopic videos for different displays is a challenging issue. Through a user study, we have shown that there is indeed a need to customize 3D videos for different displays. We have also shown that the amount of manipulation is dependent on the display size. In Chapter 3, we presented a novel system that supports adaptive streaming of 3D videos to heterogeneous receivers while accounting for display sizes, 3D video representations, and viewers’ preferences. Our system employs the DASH protocol for adaptive streaming and switching among different versions of 3D videos in order to improve the user viewing experience. In addition, we have proposed a new method for depth customization of 3D sports videos such as soccer, football, and tennis. Our method is computationally inexpensive and it maintains the scene structure of 3D videos.

We have assessed the performance of our system through another user study with 15 subjects viewing several 3D videos on five different 3D displays, ranging from a mobile phone to a large 55" TV set. Results showed that the proposed system significantly improves the depth perception in all scenarios. In addition, our subjective studies showed that depth preferences are very diverse among users. While some users enjoy 3D content with striking depth, others may prefer a more conservative 3D action. As a result, our system enables depth personalization. This feature allows viewer’s to adjust the depth to their own taste, and provides them with a more comfortable experience.
The lack of 3D content is another issue which hugely affects the adoption of 3D videos. 2D-to-3D conversion can be a promising solution which has received significant interest both in academia and industry. In Chapter 4, we presented a 2D-to-3D video conversion method for soccer. Prior methods cannot handle the wide variety of scenes and motion complexities present in soccer matches. Our method is based on transferring depth gradients from a 2D+depth database and estimating depth through Poisson reconstruction. In order to assure temporal smoothness, we proposed a spatio-temporal poisson formulation, which uses temporal gradients in addition to the spatial. Creating a 2D+depth database for outdoor sports events can be challenging. However, our key observation was that synthetic data extracted from computer games can provide a large, diverse, and accurate database for this purpose.

We implemented the proposed method and evaluated it using real and synthetic sequences. Results showed that our method can handle a wide spectrum of video shots present in soccer games, including different camera views, motion complexity, occlusion, clutter and different colors. Participants in our subjective studies rated our created 3D videos Excellent, most of the time. Experimental results also showed that our method outperforms state-of-the-art objectively and subjectively, on both real and synthetic sequences. Our method is domain specific and requires a representative database to achieve high quality results. However, we tested our method on other field sports such as baseball, field hockey, and tennis to show its potential. While our database only contained synthetic data from soccer games, ratings for non-soccer videos were promising, ranging from Good to Excellent.

Capturing sports events in 360 is also challenging. Current solutions require upgrading the entire production pipeline. Such upgrade, includes expensive equipment and special training required for setting up and operating such solutions, which causes it to be very time-consuming and costly. In Chapter 5, we proposed an alternative approach that produces VR content using the traditional broadcast setup through a post-processing stage. Our approach utilizes the rotation of the main camera to create a wide-view background panorama, which the field and players will be then overlaid on. However, due to the limited scope of the camera not all players are covered by the main camera at every instance of time. To avoid players from appearing and disappearing, we proposed to copy the players missing in the main feed from other camera feeds. For this, all feeds need to be aligned with the main camera frame. This, however, is challenging due to the huge distance between the cameras in a typical broadcasting setup, which causes large amounts of parallax between different views. To overcome this issue, our technique first obtains the relative camera parameters and estimates the 3D position of each pixel. It then removes the parallax by warping each feed to the position of the main camera.

We tested our method using three different sports games using three regular cameras located at the left, middle, and right side of the field. Subjective studies showed that our technique can generate VR content comparable to the originally captured 360 video while not introducing any major artifacts.
6.2 Future Work

The work in this thesis can be extended in multiple directions. In Chapter 3, the focus of our depth customization method is on field sports and scenes that can be well approximated by a plane. In order to support a wider range of 3D content, depth customization methods for non-sports videos need to be designed. However, it should be noted that each type of content may have its own special features and properties, which should be taken into account. For example, in sports the straightness of the lines is very important, while it might not be much of a concern in a movie.

In Chapter 4, we used a synthetic database extracted from the FIFA 13 video game to perform 2D-to-3D conversion for soccer. While our results showed the potential use of our method on other field sports, a high-quality conversion may require creating larger synthetic databases. In addition, our method can be extended to non-sports videos as well, but a proper and representative database should be designed according to the type of content.

Another extension for our 2D-to-3D conversion method can be improving its robustness to illumination variation. While our block-based matching has a degree of robustness to illumination variation as it uses SIFT, player detection can get erroneous. This is because player detection requires color modeling for field detection, and thus variations in the illumination can impact our field detection.

In Chapter 5, future work can address better handling of players in cluttered regions. Current results can be temporally inconsistent in such cases. This is because in such regions our object masks cannot distinguish the players from each other, causing the estimation of their 3D pixel positions to be erroneous. A more advanced player segmentation can be investigated for better performance. Our VR content generation technique does not rely on the availability of auxiliary data such as ground-truth camera positions and accurate depth information. However, having access to such information can further improve the results. In closed environments, depth information can be captured using infrared cameras, however, its accuracy needs to be tested.

In addition, although the static background didn’t damage the immersive experience of our generated VR content and only marginally affected the sense of presence, exploring ways of capturing some of the fans dynamics and including it in the panorama without too much complexities may enhance the results.

While our VR content generation technique focused on sports events, it may also be extended to other domains such as concerts or theatre. Generally, any event with a specific region of interest (such as the field in sports games) and a camera setup that involves multiple cameras around its region of interest may potentially benefit from our technique. However, such extensions need to be tested, and further adjustments may be required according to the selected domain.

3D and 360 videos each immerse the users into the action using a different dimension. 3D videos create the illusion of a third dimension, while 360 videos allow users to see in all directions. An interesting combination of both is presented in stereoscopic 360 videos. Similar to regular stereoscopic videos, a stereoscopic 360 video includes two 360 videos, one for each eye. Stereoscopic
360 provides a more immersive experience that includes depth perception in all direction. Recently, camera manufacturers are producing stereoscopic 360 cameras to offer users a heightened immersive experience. These cameras are typically a lot more expensive than normal 360 cameras as they require almost double the number of cameras to be installed in the 360 rig. In our work, we addressed depth customization and generation for regular 3D content, in addition to the generation of regular 360 content. Therefore, it would be interesting to explore merging all these 3 contributions to generate stereoscopic 360 content for sports, without the need for expensive stereoscopic 360 cameras. To do so, first a 360 video should be created using broadcasting cameras around the field. Next, depth should be estimated for the whole 360 panorama, and additional customizations may be required according to the rendering device and personal preferences.
Bibliography


94


