# Assignment for the course Automated Planning Theory and Practice

Davide Lusuardi
*Department of information engineering and computer science*
*University of Trento*
Trento, Italy
davide.lusuardi@studenti.unitn.it

***Abstract*—This report is intended to present and analyze the assignment for the course *Automated Planning Theory and Practice* ?? and discuss some design choices regarding its modeling and implementation.**

**The assignment is inspired by an emergency services logistics scenario where a set of robotic agents have the task to deliver crates containing emergency supplies to some injured people.**

**The assignment is divided into four subproblems, each building on the previous one with increasing complexity. In the first problem, the robot can pick up and move just one crate at time. In the second problem, the complexity increase a bit given that the robot exploits a carrier that can load up to four crates. In the third problem, is required to use durative actions in order to assign reasonable duration to different actions and model which actions can be executed in parallel. Lastly, the third problem has to be implemented within the `PlanSys2` framework, executing in a simulated environment the plan.**

## I. Introduction

Planning is a branch of Artificial Intelligence that seeks to automate reasoning about formulating a plan to achieve a given goal in a given situation. Planning is a model-based approach: a planning system takes as input a model of the initial state, the actions available to change it, and the goal conditions and produces as output a plan that will reach the goal from the initial situation.

The Planning Domain Definition Language (PDDL) is a formal knowledge representation language designed to express planning models. Developed by the planning research community, it has become a de-facto standard language of many planning systems.

The purpose of the assignment is two-fold. First, model the given scenario using the PDDL language and find a plan using a state of the art planner. Second, leveraging the PlanSys2 **??** infrastructure, integrate the model within a robotic setting.

The proposed scenario **??** is inspired by an emergency services logistic problem: a number of injured people are situated at known locations and one or more robotic agents have the task to deliver crates containing emergency supplies to each person. The planning system should formulate a plan for the robotic agents in order to deliver all the crates needed by the injured people.

The document is structured as follows: after this introduction, in Section II we explain in details our understanding of the proposed scenario and the four subproblems; in Section III we describe the design choices and the proposed solution to the problem, making some assumptions not written in the assignment document in order to constrain the problem; in Section **??**, we analyze and criticize the achieved results and briefly describe the content of the submitted archive; in Section **??**, we make some conclusions and future works are proposed.

## II. Understanding of the problem

The assignment proposes four subproblems within the same scenario, each building on the previous one.

The proposed scenario consists of an emergency situation in which there are a number of injured people who need emergency supplies. Each injured person is located in a specific place and may need some kinds of emergency supply (e.g. food, medicine, beverage, ...). There may be more people at the same location and some locations may be empty. Each crate is at a specific location and contains just one specific kind of emergency supply. Some people may already have some crates and some others may not need any crate. People don't care which crate exactly they get, only the content type is important. There can be one or more robotic agents that cooperate in order to deliver crates to people. Each location is connected to every other location, so the robotic agent can move directly between them. Initially, the robotic agents and the crates are situated at the depot, where there are not injured people.

The goal consists in deliver all the required crates to injured people.

The above considerations and the goal are common for all the four subproblems that we are going to discuss.

### A. Problem 1

This is the easiest problem. We have that a single robotic agent is present at the depot. It can perform the following actions: pick up and load just one crate that is at the same location; move to another location, moving also the loaded crate if present; deliver the loaded crate to a person that is at the same location of the robot.

### B. Problem 2

In this subproblem, the robotic agent can move crates in an different way. We introduce the concept of carrier that can be loaded with up to a specific number of crates, in this case up to four. This number is problem specific and is

modeled in the problem file. At a specific location, the robot can load the carrier with crates situated in that location and can deliver loaded crates to people. The robot can move to another location bringing a carrier with it or not.

The difficult part consists in modeling correctly the carrier and its variable capacity.

### C. Problem 3

In this subproblem, it's required to make use of durative actions. Reasonable durations should be assigned to actions coherently with their time spans and the possibility to execute actions in parallel should be analyzed taking into account real constraints.

### D. Problem 4

In this subproblem, it's required to integrate within the PlanSys2 infrastructure the last subproblem.

## III. Design Choices

In this section we discuss some design choices and some assumptions that have been made analyzing the scenario and the different subproblems: further assumptions are required in order to define better the proposed problem.

We can start saying that each crate can have just one specific content, each person can need none or more emergency supplies and can initially have some crates. This implies that each person could require more crates. We assume that a single crate with specific content is enough to satisfy the need of a person of that emergency supply, i.e., if a person needs some food, it is sufficient to deliver it just one crate containing food, no more. For this reasons, the problem file should be defined in a way that does not introduce any form of inconsistency: for example, we should not have the situation in which a person initially has a crate containing food and at the same time the person needs food.

We assume that a crate already delivered is no more available (TODO), even if it is redundant.

Initially, we need to consider the main object types used in the proposed scenario. We can simply spot the following types: `robot`, `person`, `crate`, `location`, `content`, and, from the Problem 2 on, the assignment introduces the concept of `carrier`. Each type can be easily understood without further explanations. We can note... TODO

We can now start describing the modeled PDDL predicates, actions and functions.

In order to model the location of robotic agents, people and crates, appropriates predicates should be defined. We could simply use just one predicate to express objects location, but we preferred to define a predicate for each type of objects to improve clarity. In this way, we have defined the predicates `(robot_at ?r - robot ?l - location)`, `(person_at ?p - person ?l - location)` and `(crate_at ?c - crate ?l - location)`.

As we have said, people need specific emergency supplies. To model this fact, the predicate `(need ?p - person ?co - content)` has been defined. The predicate `(have ?p - person ?co - content)` is used to define that a person has a crate with that content. In order to know if a crate has already been delivered, the predicate `(available ?c - crate)` is used: we assume that when a crate is delivered is no more available to be picked up or moved. This assumption is not strictly required, even if it sounds reasonable, and could be omitted in the modeling.

To model that a robot can be empty or can hold a crate, the predicates `(empty ?r - robot)` and `(hold ?r - robot ?c - crate)` are respectively used. We note that the predicate `empty` is not strictly required but it is useful to simplify the modeling and increase efficiency of the planner.

Three different actions have been modeled: `pick_up`, `move` and `deliver`.

The action `pick_up`, as the name suggests, models the robot that picks up a crate: the robot and the crate should be at the same location, the robot should be empty and the crate available.

The action `move` indicates the movement of the robot from one location to another.

The action `deliver` indicates the delivering of a crate to a person: the person should need emergency supplies of the same type of the content of the crate, the robot holds the crate and it is at the same location of the person.

From Problem 2 on, the introduction of the carrier, poses the need to change a bit the model introducing the `carrier` type and the `(carrier_at ?ca - carrier ?l - location)` predicate. Instead, the predicates `hold` and `empty` have been removed in favor of the predicate `(load ?ca - carrier ?c - crate)` which indicates that the crate is loaded into the carrier. In order to model the variable capacity of the carriers, the function `(capacity ?ca - carrier) - number` has been defined: the function is used to check to remaining capacity of a carrier before load a crate. The action `move_carrier` has also been added and indicates the robot that moves a carrier. Instead, the action `move` indicates now the movement of the robot without bringing a carrier with it. This action could be useful, for e.g., if we want to model the fact that the carrier is not initially at the depot.

In Problem 3, the actions have been transformed in durative actions, specifying appropriate durations. A new predicate `(busy ?r - robot)` has been added in order to help modeling which actions can be executed in parallel. We decided to model the fact that a robot cannot perform the same action or whatever any other action in parallel. This does not imply that actions cannot be executed in parallel, in fact, more robots can perform actions at the same time. Proper conditions and effects of actions have been implemented in order to enforce this situation.

In Problem 4, TODO

## IV. Results

In this section, we present and analyze the obtained results regarding the execution of the planners on the PDDL files.

The submitted archive contains the solution to the presented problems and it is organized as follows. We have created four folders, one for each problem, that contain the PDDL domain file and one ore more PDDL problem files. For each problem file it is reported in a separate file the plan found by the planner.

### A. Problem1

For this problem, we have modeled two PDDL problem files with increasing complexity. The planner used to found the solution is `Downward` using the option `--alias lama-first`.

In the first problem modeled, there are 4 people, 5 crates, 4 locations and the planner successfully found an optimal solution that requires the execution of 15 actions.

The second problem modeled is a bit more complex: the number of objects modeled increase to 8 people, 8 crates and 5 locations. Also in this case, the planner successfully found an optimal solution that requires the execution of 27 actions.

### B. Problem2

For this problem, the planner used is `Enhsp-2020` that is contained in `Planutils`.

The modeled PDDL problem contains 6 people, 6 crates, 4 locations and just one carrier with capacity 4. Executing the planner with option `-anytime`, it managed to find an optimal solution that requires performing 16 actions.

### C. Problem3

For this problem, the planner used is `Optic` that is contained in `Planutils`.

It has been decided to model two PDDL problem files. In the second one, there are two robots and two carriers while in the first one only one of both. In this way, as previously explained, only in the second problem the actions can be perfomed in parallel.

The planner managed to find a plan of duration 34 seconds for for the first problem and a plan of duration 24 seconds for the second one. As can be seen in the second plan, some actions are performed simultaneously as desirable.

## V. Conclusions

## VI. Old

Planning is the model-based approach to autonomous behavior where the agent behavior is derived automatically from a model of the actions, sensors, and goals. the main challenges in planning are computational as all models, whether featuring uncertainty and feedback or not, are intractable in the worst case when represented in compact form. In this book, we look at a variety of models used in AI planning, and at the methods that have been developed for solving them. e goal is to provide a modern and coherent view of planning that is precise, concise, and mostly self-contained, without being shallow. For this, we make no attempt at covering the whole variety of planning approaches, ideas, and applications, and focus on the essentials. e target audience of the book are students and researchers interested in autonomous behavior and planning from an AI, engineering, or cognitive science perspective.

Nowadays, sport broadcasting plays a large role in current society. Therefore, it is important to provide a high quality and visually pleasing reporting of sports events. One common issue in sports is that incidents tend to be over very quickly. Slow-motion replays can be used to illustrate these incidents as clearly as possible to the viewer. Although time is stretched in these replays, there is no exploration of the spatial scene information, which is usually important for understanding the event. A system that allows a replay from any angle adds a lot of value to the viewer experience.

Free-viewpoint video (FVV) is one of the recent trends in the development of advanced visual media type that provides an immersive user experience and interactivity when viewing visual media. Compared with traditional fixed-viewpoint video, it allows users to select a viewpoint interactively and is capable of rendering a new view from a novel viewpoint [1]. Examples of physically impossible camera views which could be desirable are the goal keepers view, a player tracking camera or even a ball camera. FVV has been a research topic in the field of computer vision since a virtualized reality system was developed [2], ranging from static models for studio applications with a fixed capture volume, controlled illumination, and backgrounds [3] to dynamic object models for sports scenes. Live outdoor sports such as soccer involve several additional challenges for both acquisition and processing phases. The action takes place over an entire pitch and video acquisition should be done at sufficient resolution in order to do analysis and production of desired virtual camera views.

In this paper, we briefly present and compare the following methods to accomplish free-viewpoint video visualization for soccer scenes: the *iview* system [3], where automatic online camera calibration, segmentation and 3D reconstruction is performed, the work of Goorts et al. [1], where a plane-sweep approach is used, and the work of Ohta et al. [4], where a billboard-based representation is employed to simplify the 3D model.

In the field of computer vision, the techniques for synthesizing virtual view images from a number of real camera images have been studied since the 1990s. Free-viewpoint video in sports TV broadcast production is a challenging problem, involving the conflicting requirements of broadcast picture quality with video-rate generation. FVV techniques for generating novel viewpoints using a multiview camera setup can be categorized into two classes [1]: 3D reconstruction and image-based rendering.

Using 3D reconstruction, it is possible to construct 3D models of objects that provides a geometric proxy, which can be used to combine observations from multiple views in order to render images from an arbitrary viewpoint. Several approaches have been realized for reconstruction: visual-hull, photo-hull, stereo and global shape optimisation [3]. The quality of the virtual view image generated by these methods depends on

the accuracy of the 3D model. In order to produce an accurate model, a large number of video cameras surrounding the object should be used. Although 3D reconstruction is robust, artifacts such as ghosting objects can be introduced if a lot of objects are in the scene.

Image-based methods generate the image of the novel viewpoint directly without explicitly reconstructing the 3D scene structure. The quality of rendered views depends on the accuracy of alignment between multiple view observations [1], [3]. Usually, these methods are limited to rendering viewpoints between the camera views.

## VII. Design Choices

In this section, we present the DTI-funded collaborative project *iview* [5], a free-viewpoint video system which enables the production of novel desirable camera views. This system exploits the already placed live TV broadcast cameras as the primary source of multiple view video. Usually, soccer matches are covered by 12-20 high-definition cameras placed all over in the stadium providing wide-baseline views. Match cameras are manually controlled to follow the game play zooming in on events. However, only a fraction of these are focused on specific events of interest and can be used for production of free-viewpoint renders, the remaining cameras cover the pitch, crowd and coaches.

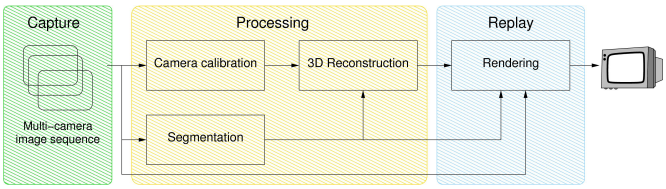


Figure 1. Overview of the *iview* free-viewpoint video system [3].

The *iview* system is composed of three main modules as shown in Figure 1: capture, processing and replay module.

Capture is performed using time synchronised acquisition from both auxiliary and match cameras. The minimal number of cameras is about four, but for good quality results a higher number is required. Camera synchronisation is achieved using standard genlock process.

The processing module computes a 3D model of the scene. This is done using segmentation of objects from the background and 3D reconstruction [6].

To allow the use of footage from match cameras and to avoid the need for prior calibration, automatic calibration of all cameras is performed using a line-based approach against the pitch lines of the captured footage, achieving a root-mean-square error of 1-2 pixels for moving cameras. The calibration is very fast and robust, capable of real-time operation for use during live match footage. Calibration estimates the extrinsic and intrinsic parameters of each camera including lens distortion.

The segmentation is needed to separate the foreground, i.e., the players, from the background. Matting of players from the green pitch is performed using chroma-keying matting. The authors developed and tested a k-nearest neighbour approach for chroma-keying and evaluated two other known techniques, *Fast green subtraction* in RGB colour space and keying in HSV colour space. The k-nearest neighbour classifier is controlled by a GUI where the user has to click on position in the image that corresponds to the background. The process is repeated until the resulting segmentation is satisfying. A deeper explanation is present in the paper [6] where the authors present and evaluate also *Fast green subtraction* and keying in HSV.

The accumulation of errors from calibration and matting can cause large errors in the reconstruction of the scene, such as loss of limbs. Therefore, robust algorithms have been developed for scene reconstruction. One possible technique is called visual-hull (VH) and represents the maximum volume occupied by an object given a set of silhouettes from multiple views [7]. The visual-hull is a single global representation integrating silhouette information from all views. A polygonal mesh surface is typically extracted and texture mapped by resampling the captured multiple view video for rendering [8]. Due to accumulating errors in camera calibration and segmentation, visual-hull accuracy is reduced. A refinement of the view-dependent visual hull (VDVH) [9], using stereo correspondence to interpolate between captured views, can be used to overcome these issues, achieving the best alignment between adjacent views and hence improving visual quality. More information about *iview* 3D reconstruction can be found in [3], [6], [8].

Finally, the replay module renders the novel view of the scene using the computed 3D model together with the original camera images. Cameras closer to the virtual viewpoint are chosen to generate the novel viewpoint.

The method proposed achieves an image quality comparable to that of the input images and it is robust to the wide-baseline moving camera views at different resolutions. Calibration and segmentation are very important in order to obtain an overall good quality of the system. Degradation in image quality will also occur if there are insufficient views for reconstruction.

## VIII. Image-based rendering - extended plane sweeping

In this section, we present the work of Goorts et al. [1], i.e., an image-based approach to generate virtual camera view interpolating real camera images. Instead of performing 3D reconstruction, image-based methods directly generate the image of the novel viewpoint. When multiple cameras are present, plane sweeping can be used [10] for both small and wide baseline setups. Plane sweeping has already been used for novel viewpoint in soccer scenes. Goorts et al. [11] present a method with two plane sweeps and a depth filtering step suitable for smaller baseline setups of about 1 meter. This method presents some problems like disappearing players when they overlap in the image.

The method presented here is fully automatic and employs GPU parallel processing to achieve fast processing speed. The system setup consists of 7 static cameras placed in a

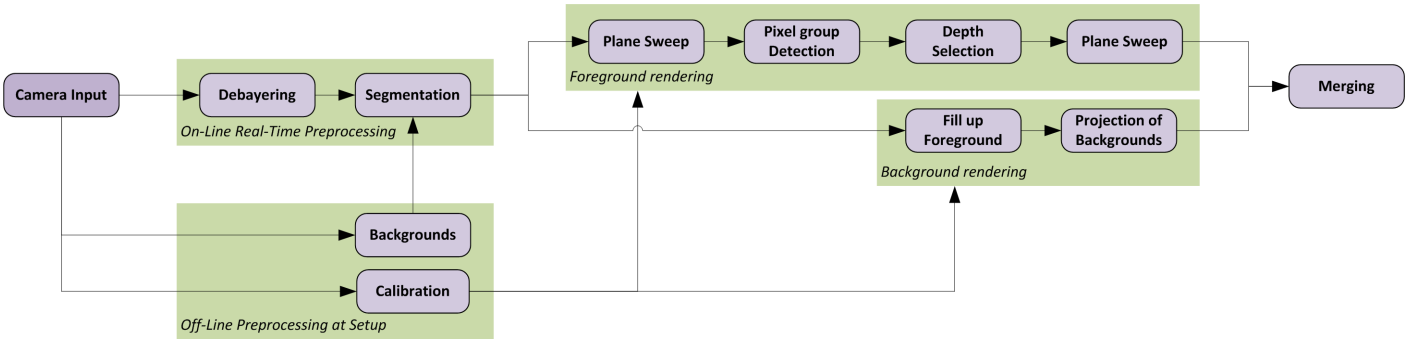


Figure 2. Overview of the extended plane sweeping method. Both the non real-time and real-time phase are shown [1].

wide baseline setup, i.e., 10 meters between each camera. All cameras are synchronized on shutter level using a global clock. The generation of novel viewpoint consists of two steps as shown in Figure 2: a first off-line preprocessing phase and a real-time interpolation phase.

The preprocessing phase is responsible for camera calibration and background determination. Cameras are calibrated in order to acquire their position, orientation and extrinsic and intrinsic parameters. The calibration method of Svoboda et al. [12] has been employed for this purpose: SIFT features are extracted from a number of frames and the pairwise matching between them is calculated using the k-d tree algorithm; these matches are tracked across different image pairs obtaining point correspondences between multiple images [1]. The background of every image stream is determined using a per-pixel median approach applied to about 30 images per stream (2 seconds apart each other).

The real-time phase generates images for a chosen virtual camera position and a chosen time in the video sequence. First, camera images are debayered and segmented. These images are then used to process foreground and background independently. The foreground rendering uses a plane-sweep approach followed by depth filtering and a depth-selective plane sweep as explained in [1]. The foreground and background are then merged together according to the segmentation information.

Debayering consists of converting the raw images to its RGB representation. Segmentation is based on backgrounds obtained during off-line preprocessing and is performed on a per-pixel basis using the differences between the color values and three thresholds. This allows fast segmentation in high quality.

Applying this method, the authors obtained high quality results using wide baseline setup and typical artifacts of normal plane sweeping, such as ghost players, are removed. Some other artifacts can still be present, such as ghost lines, caused by the simplified assumption of the geometry of the pitch.

## IX. Billboard-based visualization

In this section, we present the work of Ohta et al. [4]. The authors use billboard representation to make a 3D model of each player. This method is simpler than full 3D reconstruction and requires less computation. A player billboard is a small rectangle standing perpendicular to the ground and a 2D texture is shown on it. The difference between 3D reconstruction and billboard representation is shown in Figure 3: as we can see, the visual difference is clear at a close viewpoint but becomes very small at a distant one. For this reason, becomes particularly important to place player billboards at a right place and direction.

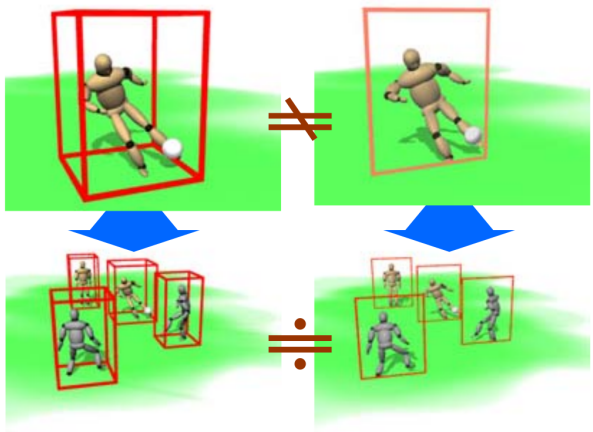

Figure 3. Appearance similarity between 3D reconstruction and billboard in close and distant view [4].

The system proceeds as follows: first it extracts texture segments from camera videos, then selects appropriate textures according to the virtual viewpoint and finally layouts the player billboards in virtual space.

Texture extraction phase consists in obtaining the location of each player and extracting texture segments from every image video by projecting player location onto the image plane. Background region is removed in the texture by video capturing PC. Please note that camera calibration is done before this phase.

Texture selection phase selects a set of texture to be sent to each viewer based on his viewpoint. Given a viewpoint, the system finds the camera that minimizes the angle between the line from the viewpoint to the player location and the line from the camera to the player location. Then, a texture

segment obtained by that camera is selected and placed so that the texture faces the viewpoint [4].

One problem may happen when players are overlapped each other at a certain camera and billboard texture might include both of them. To eliminate extra player region, authors used stereo based method [13] as explained in [4].

## X. Conclusion

We have briefly presented three free-viewpoint systems for soccer games: the *iview* system, an image-based method using plane sweeping and a billboard-based method.

The *iview* system is a robust method using a multi-camera setup that performs 3D reconstruction exploiting a refinement of the view-dependent visual hull method.

The image-based method presented interpolates real camera images using a plane sweeping approach. High quality images using wide baseline setup are obtained and typical artifacts are removed.

The billboard-based method is simpler than full 3D reconstruction and requires less computation. However, this method produces lower quality images and some visual artifacts can be present.

## References


[1] P. Goorts, S. Maesen, M. Dumont, S. Rogmans, and P. Bekaert, "Free viewpoint video for soccer using histogram-based validity maps in plane sweeping," in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 3, 2014, pp. 378–386.

[2] T. Kanade, P. Rander, and P. J. Narayanan, "Virtualized reality: constructing virtual worlds from real scenes," *IEEE MultiMedia*, vol. 4, no. 1, pp. 34–47, 1997.

[3] A. Hilton, J.-Y. Guillemaut, J. Kilner, O. Grau, and G. Thomas, "Free-viewpoint video for tv sport production," in *Image and Geometry Processing for 3-D Cinematography*, 2010.

[4] Y. Kameda, T. Koyama, Y. Mukaigawa, F. Yoshikawa, and Y. Ohta, "Free viewpoint browsing of live soccer games," in *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, 01 2004, pp. 747–750.

[5] "The iview project," http://www.bbc.co.uk/rd/projects/iview.

[6] O. Grau, G. Thomas, A. Hilton, J. Kilner, and J. Starck, "A robust free-viewpoint video system for sport scenes," in *2007 3DTV Conference*, 06 2007, pp. 1 – 4.

[7] Heung-Yeung Shum, Sing Bing Kang, and Shing-Chow Chan, "Survey of image-based representations and compression techniques," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 11, pp. 1020–1037, 2003.

[8] O. Grau, A. Hilton, J. Kilner, G. Miller, T. Sargeant, and J. Starck, "A free-viewpoint video system for visualisation of sport scenes," 2006.

[9] G. Miller and A. Hilton, "Exact view-dependent visual hulls," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 1, 2006, pp. 107–111.

[10] R. Yang, M. Pollefeys, H. Yang, and G. Welch, "A unified approach to real-time, multi-resolution, multi-baseline 2d view synthesis and 3d depth estimation using commodity graphics hardware." *Int. J. Image Graphics*, vol. 4, pp. 627–651, 10 2004.

[11] P. Goorts, C. Ancuti, M. Dumont, S. Rogmans, and P. Bekaert, "Real-time video-based view interpolation of soccer events using depth-selective plane sweeping," *VISAPP 2013 - Proceedings of the International Conference on Computer Vision Theory and Applications*, vol. 2, 04 2013.

[12] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multi-camera self-calibration for virtual environments," *Presence*, vol. 14, pp. 407–422, 08 2005.

[13] T. Koyama, I. Kitahara, and Y. Ohta, "Live mixed-reality 3d video in soccer stadium," in *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.*, 2003, pp. 178–186.