Automatic Camera Selection for Format Agnostic Live Event Broadcast Production

Rene Kaiser, Wolfgang Weiss, Gert Kienast,
Werner Bailer, Marcus Thaler & Georg Thallinger

JOANNEUM RESEARCH
DIGITAL – Institute for Information and Communication Technologies

Graz, Austria

{firstname.lastname}@joanneum.at

Abstract

The FascinatE broadcast production system is based on a format-agnostic approach. It produces for a range of output devices in parallel. Its 180° panoramic camera records the complete scene throughout, from a single point of view. Viewers may explore the scene freely by panning and zooming though the scene. However, they might also lean back and enjoy a personalized stream which the Production Scripting Engine (PSE), the system's Virtual Director, produces while taking the viewer's individual preferences and playout device capabilities into account. We present our distributed PSE implementation, and discuss its research challenges and limitations of the current solution.

1 Introduction

The FascinatE[1] project investigates a novel approach for live event broadcast production. A number of innovative features aim to enhance the *immersiveness* of viewer experience and save production cost.

The FascinatE system (Schreer 2011) follows a format agnostic approach which means that it does not produce for a specific output format. The audio-visual scene is captured in very high quality, which is achieved by using the Fraunhofer HHI OMNICAM (Weissig et al., 2012), a 180° ultra-high definition panoramic camera that continuously captures the whole scene with a

1 http://www.fascinate-project.eu/

resolution of 7k by 2k pixels. The same approach is followed for audio: an Eigenmike, a spherical microphone that contains 32 individual microphone capsules, is used to capture the entire sound field instead of individual channels.

All these content streams are represented in the so-called *Layered Scene Representation* from which formats for specific playout devices can be derived. Screens range from mobile phones and regular HD television sets to panoramic cinema screens and different loudspeaker setups. This acquisition system has been used in three demonstration productions in the domains of soccer, dance performance and an orchestral performance of classical music. See example screenshots of the panoramic video in Figure 1.

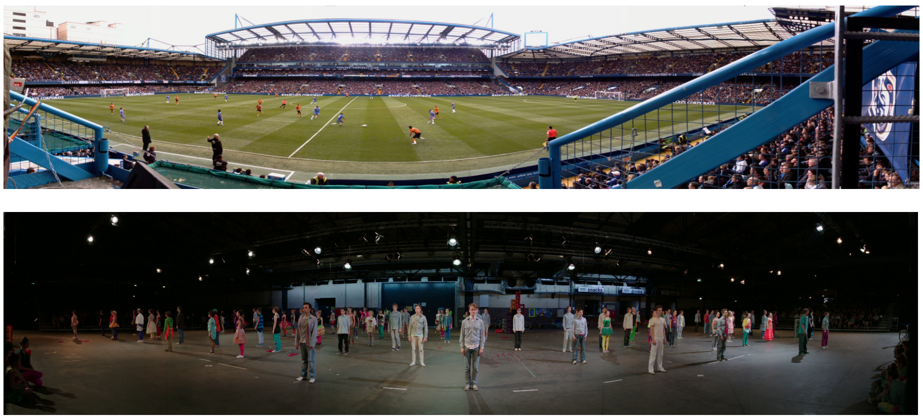

Figure 1 Panoramic screenshots from two production domains: a soccer match Chelsea F.C. vs. Wolverhampton Wanderers (2011) and below a dance performance directed by Sasha Waltz with the Berlin Philharmonic Orchestra (2012)

As the huge amount of content poses challenges for transmission of the stream, FascinatE also investigates approaches of tiled streaming (Niamut et al., 2011) which transmit only the part of the panoramic stream which is currently viewed. For interactive navigation in the panorama on the end-user side a gesture-based interface is developed providing the possibility to control the broadcast with intuitive gestures for e.g. zooming, panning, pausing or adjusting the volume (cp. Suau et al., 2012). The content streams are automatically analysed for semantic annotation, e.g. detection of persons and salient regions in the visual domain as well as audio events in the audio stream. An operator can manually define relevant events for a production (e.g. a foul or a goal).

The following focuses on the Production Scripting Engine (PSE), a software component aiming at camera selection automation. It decides where to position virtual cameras (picture-in-picture), how to frame static and dynamic shots, and when to cut, in order to select the most relevant action in the scene while respecting cinematographic principles. It fuses the aforementioned inputs and produces live content streams for different playout devices in parallel, enabling new forms of content consumption beyond personalization. In contrast to classic TV, different viewers may watch different parts of the scene, framed in the style of their choice. A shot in our context is a synonym for a virtual camera.

2 Automating Camera Selection and Framing

The Production Scripting Engine is a distributed component that takes decisions on automatic camera selection. It continuously decides what is visible and audible at each playout device, taking individual preferences of the viewers and capabilities of their devices into account. A metaphor for such systems is *Virtual Director*. The PSE's research problem of automatic camera selection and framing is multifaceted. For our prototype implementation, we decided to take a rule-based approach. The PSE's behaviour is defined as a set of production rules, which a rule engine can execute. Automatic execution of cinematographic principles has also been investigated in the domain of videoconferencing (Kaiser et al., 2012) as well. Our aim is to work towards a generic framework that can be adapted to different production system increments, and also to different production genres. In that realm, we investigate to which degree production grammar re-use is feasible across genres and how it can be supported by tools. As a central part of the FascinatE system, the PSE interfaces with a range of other components, as illustrated in Figure 2.

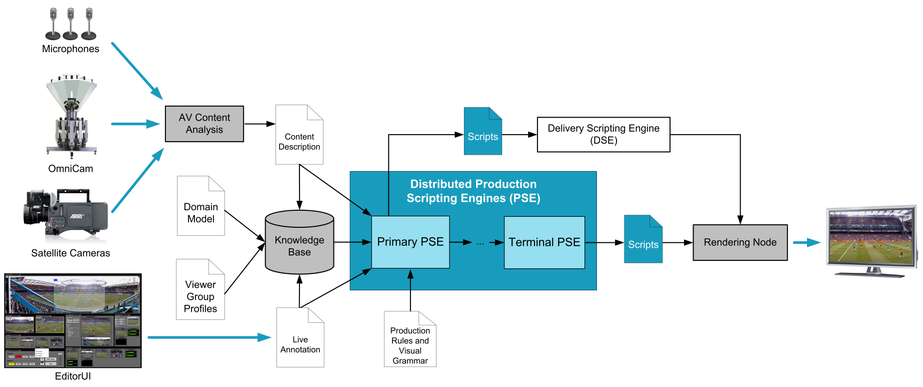

Figure 2 Excerpt of the FascinatE system architecture illustrating the interplay of components relevant to automatic camera selection (Production Scripting Engines)

The PSE software component is distributed to form a chain through the production network. The minimal configuration consists of two instances, as illustrated by Figure 3. The primary PSE instance runs at the production end, which has the following specific tasks:

- The primary PSE processes the real-time stream of low-level events as extracted by A/V content analysis;
- It is integrated with the Editor UI toolset, i.e. an interface for production professionals that allows manual annotation and decision intervention;
- It uses a knowledgebase for spatiotemporal queries and to retrieve meta-data for e.g. replays;
- It informs the delivery network via the Delivery Scripting Engine about its shot candidates. This information can be used for content transmission optimisation.

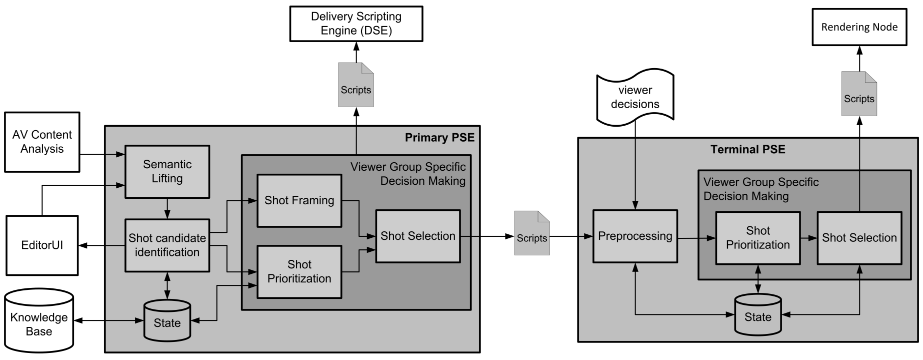

Figure 3 Internal architecture of the Production Scripting Engine in a configuration with two instances

An instance at the terminal end is also required, as it is responsible for taking final decisions and for instructing the renderer. Any number of PSE instances might be added in the middle of this chain, e.g. with the specific purpose to filter and re-prioritise shot candidates with respect to a certain aspect, e.g. privacy or content rights. Editing decisions will be updated and restricted down the PSE chain, and in addition to a list of prioritised shot options, metadata is passed from one instance to another. These messages, as well as the final instructions for the renderer, are called *scripts*.

While traditional TV broadcast produces a singular video stream where every viewer gets to see the same edit, one of the key advantages of the PSE is the ability to service a large number of individual preferences. A central aspect of the format-agnostic vision is to realize personalized A/V streams that respect the viewers' connection and device capabilities, and their domain-dependent preferences. Examples for the latter are selection between several cinematographic styles, focus on certain persons, groups, or types of actions. This automatic process is informed by the content analysis algorithms and a set of dedicated production tools that allow manual support of the decision making sub-processes. The Editor UI toolset is designed to enable basic features such as manual annotation of higher-level concepts, including properties such as their location within the video panorama and their temporal extent.

The PSE is based on an event processing engine, mainly for performance reasons, as it is required to react to input and to take decisions in real-time. Most of its logic is executed by a rule-engine. The PSE's behaviour is defined by a set of domain-dependent production principles, implemented in a format specific to a rule engine. These principles define how the PSE is automatically framing virtual cameras within the omni-directional panorama, how camera movements are smoothed, when cuts and transitions to other cameras are issued etc. Further details on the PSE's approach and architecture can be found in (Kaiser, Weiss & Kienast, 2012).

3 Production Grammar

The following discusses the set of production rules which the Production Scripting Engine executes, the *production grammar*, which is partially domain-specific and partially generic. The rules define both the pragmatic scope of how to capture domain-dependent actions, and the cinematographic scope how to do that in a visually aesthetic manner. However, a key issue is that, in general, production rules are not independent. The engineering effort necessary to structure their interplay and to balance their effects and side-effects is considerably high. Competing and contradicting principles need to be resolved so that the desired decisions are made, which is especially challenging in this context. We aim to understand how rule re-use can be encouraged.

The PSE's behaviour consists of several sub-aspects, of which the most important are:

- Decide where in the panoramic video to place virtual cameras as shot candidates. They might me static or moving as pan, tilt and zoom, have a certain static or changing size (type of shot, e.g. close-up), are moving at a certain speed. The latter properties might depend on preferences of individual viewers.
- Decide when to drop shot candidates since the action they cover is no longer relevant.
- For each viewer (group) individually, decide at which point in time to cut from one virtual camera to another.
- Decide how to perform those cuts: depending on the location and content of the two virtual cameras, either a hard cut or a transition is decided. We do not use fades in our prototype system.
- If in addition to the OMNICAM panorama one or more broadcast cameras are available, decide when to use those sources which offer greater level of detail.

In order to define the PSE's intended behaviour; we observe TV broadcasts and interview professionals. The production rules are initially captured in natural language before they are developed as JBoss Drools rules. Naturally, rules can only be triggered by an event that is directly observable or can be derived through semantic lifting. We utilize an exchangeable domain model that defines these higher-level events, and also the primitives, the low-level events which are automatically extracted or manually annotated. The domain model also holds a domain-specific configuration which allows defining a certain style for a production.

4 Scene Understanding

Without understanding what's happening in the scene to a certain extent, the PSE's decisions to show a certain are of the panorama to a viewer could only be poor. Therefore, the first step in the workflow of the PSE is to achieve an abstract understanding of which domain-specific actions are currently happening in parts of the scene.

Two channels are informing the software system about the real world: generic automatic feature extraction modules and a generic manual annotation/monitoring toolset. Their input is transmitted as MPEG-7 AVDP (2012) documents. The PSE will subsequently process a real-time stream to bridge the semantic gap between low-level annotations and higher-level concepts which are part of the production grammar, the rules and principles that define how the action is framed.

The following will explain the purpose of two content analysis models, person tracking and saliency estimation. Further subsections discuss the Editor UI toolset and the PSE's subprocess of *Semantic Lifting*.

4.1 Person Detection and Tracking

The system's most important automatically extracted feature is person detection and tracking. Our CUDA[2]-based algorithm was developed to detect and track persons in different content genres. As it operates on a single point of view, tracks are expected to break due to occlusions etc. at times. Therefore, continuous identity assignment is not possible.

Given the enormous resolution of the OMNICAM, the algorithm's real-time capability is a crucial requirement. It is not feasible to perform the A/V content analysis on the full resolution panorama on a single standard computer due to limitations regarding both the bandwidth of network interfaces and the computational requirements of the A/V content analysis itself. Person detection and tracking is therefore performed independently for each of the HD resolution tiles in parallel. For further implementation details, please refer to (Kaiser et al., 2011).

2 http://www.nvidia.com/object/cuda_home_new.html

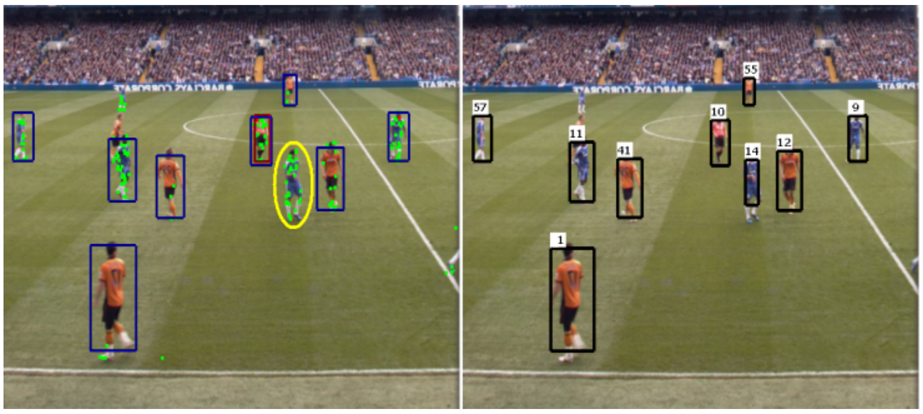

Figure 4 Result of the soccer player detection

4.2 *Visual Saliency Estimation*

The second automatically extracted feature is visual saliency estimation. In our implementation, we consider visual saliency in spatiotemporal space as a generic measure that can be applied to any content domain. It does not provide semantic meaning on its own, but rather indicates where a human observer of the scene might naturally look.

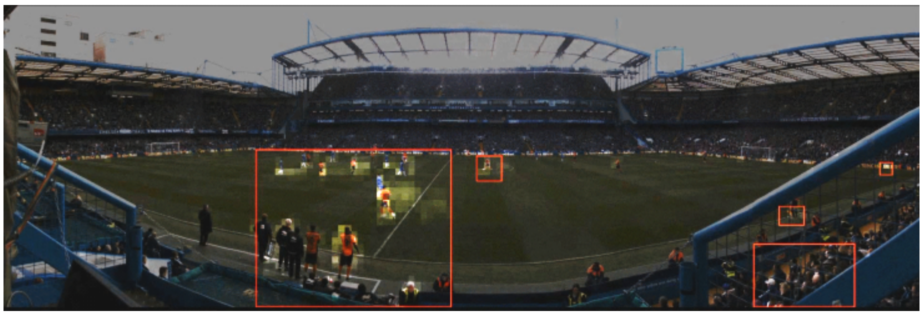
Figure 5 Regions highlighted by the visual saliency detector

We use the feature maps calculated from basic saliency measures in order to integrate intensity and colour histograms calculated in a recent time window. These reference histograms represent the content of the scene recently seen by the viewers. Significant changes in the current frame mean a new visual stimulus and indicate the regions of interest. Thus, the histogram differences between the current frame and the previously stored reference model can be used as saliency indicator. In our implementation, we divide the image area into grids of size 40 x 40 pixels. Each grid cell provides its own reference histogram with 256 bins. To build the reference model, the 20 most recent frames are used. To determine a current saliency estimate, the saliency features are matched against the reference model.

4.3 Toolset: Editor User Interface

The Editor UI can be compared to toolsets for operators (e.g. vision mixer) in traditional broadcasts. It is a professional's tool that allows the production team both steer and observe the internal behaviour of the PSE. It helps to monitor PSE behaviour by visualizing scripting decisions and decision factors. Depending on the production, the number of users working in parallel may vary.

As its main purpose, the Editor UI allows to annotate key domain-specific actions happening in the scene which the automatic components cannot detect. The Editor UI's main features concerning the PSE are:

- Annotation of actions through domain-specific concepts
- Creation of temporally available static virtual cameras and their properties
- Creation of virtual moving camera, following a panning/zooming path
- Live tracking of moving objects
- Domain-specific configuration
- Monitoring the PSE's internal shot candidates and decisions
- Re-prioritisation of shot candidates

4.3 Toolset: Editor User Interface

A key sub-process of the PSE is called *Semantic Lifting*. It deals with the problem that the incoming information about the scene is on a different semantic level than the production rules for camera selection decisions. In order to bridge this semantic gap, this component aims to achieve an *understanding* of what is currently happening. From a technical point of view, it aims to derive domain-specific higher-level concepts. It does so by looking for certain spatiotemporal constellations of low-level events in the streams as triggers. It emits a range of higher-level events to inform subsequent decision-making components. We chose to implement Semantic Lifting with

3 https://www.jboss.org/drools/

JBoss Drools[3], a hybrid rule engine that is also a Complex Event Processing (CEP) engine (cp. Etzion & Niblett, 2010). The latter is especially interesting in this case, since the advantageous processing performance of CEP helps to deal with a high number of low-level events in real-time.

5 Decision Making

Decision making is the PSE's most challenging subtask. Most of the process is parallelized, with one thread per viewer (group). The basic idea is to identify and manage a generic list of shot candidates that can be used across viewers, but prepare them specifically to the viewers' parameters. Priorities are re-calculated over the chain of PSEs, according to different aspects. The following discusses the individual subcomponents.

5.1 Shot Candidate Identification

This component creates and maintains a list of usable shot candidates, i.e. a list of real and virtual (cropped) views from the omni-directional panorama. The output are options that subsequent components use to take personalized camera decisions. Shot Candidate Identification builds on the higher-level understanding achieved by Semantic Lifting to decide which views to select as candidates, while also keeping its options balanced with regard to the diversity of what they cover. Some candidates can be directly derived from inputs of the Editor UI. The component makes use of the definition of annotation concepts and shot properties that are loaded from a domain-dependent model. The component determines at least one candidate at all times; the actual number depends on the scenario.

5.2 Shot Framing

The aim of Shot Framing is to frame shot candidates, i.e. to define bounding boxes for static and moving virtual cameras. Their size and aspect ratio correspond to the viewing device. When framing moving objects, *smooth* camera pans are required to capture their movement. Movement smoothing further has to take the panorama boundaries into account. The calculation employs a spring model for smoothing out minor movements and avoiding sudden stops, taking the object type, direction and speed of movement into account. As an example, a horizontally moving athlete is positioned side of the image centre so that more of the running direction area is seen (looking room). Further, the bounding box size depends on the distance of the object to the camera, i.e. more distant objects are covered by smaller boxes so that they appear larger.

5.3 *Shot Prioritisation*

Shot candidate prioritisation starts with the shot type for which a number of properties are defined in the domain model. For each PSE instance in the chain, the priorities are re-calculated, and due to the instance's purpose, some candidates might even be filtered out. Examples are:

- Implementing privacy rules, e.g. disallowing close-up shots on the audience
- Checking content rights, making sure viewers get to see only content they are entitled to consume
- Biasing towards certain types of actions or objects, e.g. one of two competing sports teams, favourite players etc.

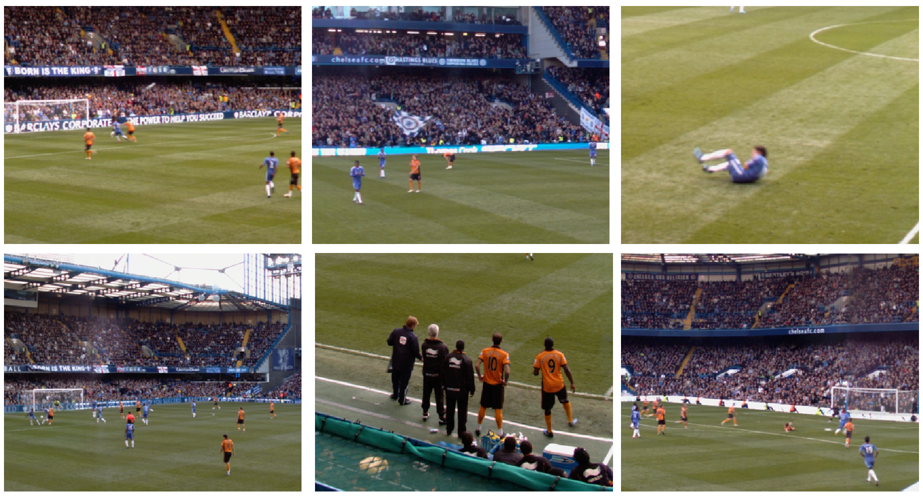
Figure 6 Example shots from the soccer domain as used by the PSE

5.4 *Decision Making*

Since the PSE's decision making mechanism is distributed, scripts contain not only decisions, but also candidates and metadata. The Decision Making component decides not only to which shots to cut, but also when. The rules may be triggered by the occurrence of higher-level events and states that are specific to each viewer group. They might also be triggered by cinematographic rules that execute e.g. that no shot can last longer than 10 seconds. Since we are dealing with a real-time broadcast system, decisions have to be taken fast and with constant delay.

6 Discussion and Outlook

We have presented several aspects of the FascinatE format agnostic production system, based on the Layered Scene Representation. In detail, we described the Production Scripting Engine (PSE), the system's *Virtual Director*, which chooses between camera views in order to (semi-)automatically compile an individual view on a FascinatE broadcast scene, respecting viewer preferences and viewing device capabilities. It follows a rule-based approach that reasons with both pragmatic and cinematographic principles. Rules are triggered by the occurrence of higher-level events and states that are specific to each viewer group. The current prototype deals with content from two very different domains, soccer and dance performances.

What we achieved so far is to develop a design for a flexible Virtual Director engine that can be easily adapted to different productions domains. We have implemented parts of the engine in the current prototype and carefully chose an approach that allows us to (a) manually define and automatically execute the PSE's desired behaviour based on a suitable formalism and (b) to make sure the overall delay of the processing chain is small enough to fulfil real-time requirements. The latter could be achieved by choosing a (forward chaining) rule-based approach combined with the advantages of a CEP engine.

Defining the set of production rules and their interplay requires engineering effort, and even though replicating the creative brilliance of experienced seems to be an impossible task, we are confident the basis now in place will allow realizing productions in acceptable quality. A limiting factor is the amount of automatic content analysis available. To make up for it, we will focus on close collaboration between the PSE and the Editor UI i.e. to assist, through manual annotations, the PSE's understanding of the current situation.

Key drawbacks of our current prototype are:

- Can't replicate the creative brilliance of human operators and directors
- Rules are reactive, however, a certain amount of prediction intelligence could improve the output significantly

- Works with one panoramic camera with fixed focus the virtual director is restricted to a single point of view and can't play with focus
- Does not take viewer interaction into account, so far only respects static preferences

So far, work on scripting has mainly been concerned with automatic camera viewpoint selection. Beyond the visual output, the viewers' QoE could be enhanced by intelligently matching an individualized audio feed with the visual viewpoint. As an example for sports broadcast, if there is an audience shot after a successful score, the audio should correspond (loud cheers). More general, a viewer may want to hear the fans of his/her favourite team more than those of the opposing. The visibility of objects should correspond to their audibility. Objects that are currently not visible may not be audible at all or at a dimmed level, depending on the distance and other objects between them and the viewpoint.

Acknowledgement

The research leading to this paper has been partially supported by the European Commission under the contract FP7-248138, "FascinatE – Format-Agnostic SCript-based INterAcTive Experience".

References

Etzion, O. & Niblett, P. (2010). *Event Processing in Action*. Manning Publications.

Kaiser, R., Thaler, M., Kriechbaum, A., Fassold, H., Bailer, W. & Rosner, J. (2011). Real time person tracking in high-resolution panoramic video for Automated broadcast Production, in: *Proceedings of the 8th European Conference on Visual Media Production (CVMP 2011).*

Kaiser, R., Weiss, W. & Kienast, G. (2012). The FascinatE Production Scripting Engine, in: *Lecture Notes in Computer Science, Volume 7131, Advances in Multimedia Modeling – 18th International Conference, MMM 2012*, pp. 682–692.

Kaiser, R., Weiss, W., Falelakis, M., Michalakopoulos, S. & Ursu, M. F. (2012). A Rule-Based Virtual Director Enhancing Group Communication, in: *2012 IEEE International Conference on Multimedia and Expo Workshops*, pp. 187–192.

MPEG-7 AVDP (2012). *Information technology – Multimedia content description interface – Part 9: Profiles and levels, AMENDMENT 1: Extensions to profiles and levels*. ISO/IEC 15938-9:2005/PDAM 1:2012.

Niamut, O. A., Prins, M. J., van Brandenburg, R. & Havekes, A. (2011). Spatial Tiling And Streaming, in: *An Immersive Media Delivery Network. Adjunct Proceedings of EuroITV 2011*, Lisbon, Portugal.

Schreer, O., Thomas, G., Niamut, O. A., Macq, J.-F., Kochale, A., Batke, J.-M., Ruiz Hidalgo, J., Oldfield, R., Shirley, B. & Thallinger, G. (2011). Format-agnostic Approach for Production, Delivery and Rendering of Immersive Media, in: *NEM Summit 2011*, Torino, Italy, 27th September, 2011.

Suau, X., Casas, J. R. & Ruiz-Hidalgo, J. (2012). Real-Time Head and Hand Tracking based on 2.5D data, in: *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 575–585.

Weissig, C., Schreer, O., Eisert, P. & Kauff, P. (2012). The Ultimate Immersive Experience: Panoramic 3D Video Acquisition, in: *Proc. 18th Int. Conf. on Multi-Media Modeling (MMM 2012)*, Klagenfurt, Austria, 2012.