Practical Augmented Visualization on Handheld Devices for Cultural Heritage

Giovanni Murru
Sapienza Univ. of Rome
giovanni.murru@gmail.com

Marco Fratarcangeli
Sapienza Univ. of Rome
frat@dis.uniroma1.it

Tommaso Empler
Sapienza Univ. of Rome
tommaso.empler@uniroma1.it

ABSTRACT

In this paper, we present a framework for the interactive 3D visualization of archaeological sites on handheld devices using fast augmented reality techniques. The user interface allows for the ubiquitous, personalized and context-aware browsing of complex digital contents, such like 3D models and videos. The framework is very general and entirely devised and built by the means of free, cross-platform components. We demonstrate the flexibility of our system in a real case scenario, namely the augmented visualization of a historically reliable model of the Ancient Forum of Nerva located in Rome, Italy.

Keywords
Virtual and augmented reality, personalized heritage visits, mobile guides, location-aware.

1 INTRODUCTION

Augmented reality (AR) is an emerging computer technology where the perception of the user is enhanced by the seamless blending between real environment and computer-generated virtual objects coexisting in the same space. The resulting mixed image supplements reality, rather than replacing it [7].

In the context of cultural heritage, augmented reality is used to blend visual representations of historical monuments, artifacts, buildings, etc., into the real environment visited by the audience (e.g., tourists, students, researchers). For example, in virtual Pompeii [15], virtual characters representing ancient Romans are blended into the real environment; the user is able to perceive them by means of an ad-hoc mobile AR system. Such applications create simulations of ancient cultures by integrating them in the actual real environment. In this way, the user can learn about the culture by directly interacting with it on site.

Augmented reality systems are rather complex and involve technologies from different areas such as computer vision, computer graphics and human-computer interaction, in order to synthesize and deliver the virtual information onto the images of reality. Generally, the system must 1) track and locate the real images, 2) display the virtual information and 3) align and superimpose the virtual data onto the real image. The main challenge in the design of these systems lies in the seamless integration of computationally-expensive software modules and energy consuming hardware in a framework that must run at interactive rate and, at the same time, be portable by the user. Traditionally, the mobile systems [19, 15] require the user to wear a set of hardware devices such as cameras, electronic compasses, small laptops, which makes the whole system not comfortable and limits *de-facto* the massive spreading of mobile AR systems in the context of cultural heritage.

The recent increase of the computational capabilities, the sensor equipment and the advancement of 3D accelerated graphics technologies for handheld devices, offer the potential to make the AR heritage systems more comfortable to carry and wear, facilitating the spread of this kind of AR systems to the mass market.

1.1 Contributions

In this paper, we present a novel mobile framework for augmented reality in the context of cultural heritage, running on modern handheld devices. The framework implements context-aware tracking, 3D alignment and visualization of graphical models at interactive rate. Using this framework, the user is free to roam around archaeological sites using non-invasive and already in use devices such as modern smartphones and tablets. The framework is composed by free, cross-platform software modules, making it easier to reproduce.

The applicability of the framework is tested by providing an augmented view of the Ancient Forum of Nerva, which was one of the Imperial Fora during the Roman Empire age. The 3D model has been designed accord-

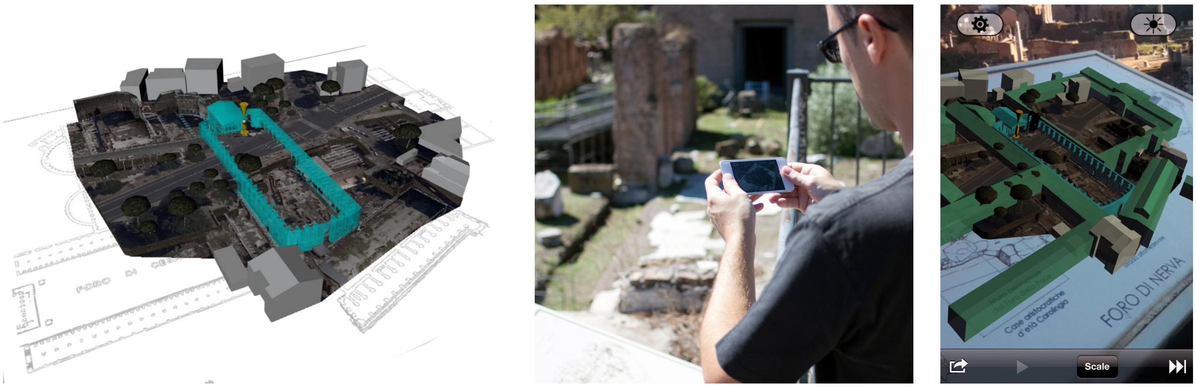

Figure 1: *Left.* 3D reconstruction of the Ancient Forum of Nerva in Rome, Italy. *Middle.* Augmented view of the archeological artifact on-site. *Right.* A screenshot of the framework running on a handheld device.

ing to the information acquired from previous archaeological studies [18].

2 RELATED WORK

Augmented Reality is a technology allowing for extending the vision of real world with superimposition of digital information, e.g. 3D virtual objects, 2D images and icons, labels, etc. Augmented reality is not intended for replacing the reality like traditional virtual reality, but it rather enhances it with digital data, making virtual and real objects coexist in the same space. In general, an augmented reality system must be equipped with display, tracker, graphics capabilities and appropriate software [6, 7, 9].

Head-Mounted Displays [13] are one of the most popular approaches for delivering mobile augmented reality in the context of cultural heritage. The Archeoguide project is among the pioneer systems for the on-site exploration of outdoor sites [11, 19]. Such a system is able to track position and orientation of the user employing a camera and an electronic compass, both mounted with the display unit. This allows for inferring the field of view and displaying 2D virtual images and information of the scene observed by the user. However, the weight and dimension of the required hardware devices makes the whole system uncomfortable to wear.

Modern handheld devices, such like smartphones and tablets, have a complete set of high quality sensors such as 3-axis gyroscope, ambient light sensor, accelerometers, magnetometer, proximity sensor, and assisted GPS; hence they are well suited for the development of augmented reality systems in a scenario similar to Archeoguide. Recent research efforts have provided several mobile solutions [8, 12, 20, 10] but are still not able to visualize context-aware, outdoor 3D models in a general manner.

Existing commercial mobile applications in the context of cultural heritage touring [3, 4, 1] lack of the capability to interactively blend 3D content to the observed real scene. In general, they completely replace the video feed coming from the camera with a 3D scene, requiring the user to stand in a known position and implementing a simple alignment based on the compass of the device, without an actual tracking and registration of the video feed with the virtual objects as it happens in augmented reality systems.

3 OVERALL DESIGN

In an augmented reality system, objects in real and virtual worlds must be precisely aligned with respect to each other, otherwise the illusion that the two worlds coexist is compromised. An augmented reality system lacking of accurate registration is not acceptable [6], and this is a requisite for establishing a connection between the features of 2D images of reality and the 3D virtual world frame [9]. Computer vision is extensively used in AR systems mostly for two main tasks: image tracking and reconstructing/recognizing. Tracking algorithms interpret camera images and, in this context, can be categorized in two classes: those relying only on feature detection and those using a precomputed model of the tracked object's features.

The framework employs the markerless tracking capabilities of the Vuforia SDK [5] to compute in real-time the position and orientation (pose) of some predefined image targets located in the real environment. Once the pose is retrieved, a virtual scene is rendered, overlaid and registered onto the video camera feed by using the OpenSceneGraph module [21].

Hence, the proposed framework is based on the communication between two main software components, namely Vuforia and OpenSceneGraph, providing a touch interface to the user (see Fig. 2). Vuforia is the module responsible for tracking an image target and estimating its pose in 3D space, while OpenSceneGraph is the one delegated to manage the rendering of the models. The most important features of these packages, and how they are employed within the framework, are briefly depicted in the following sections.

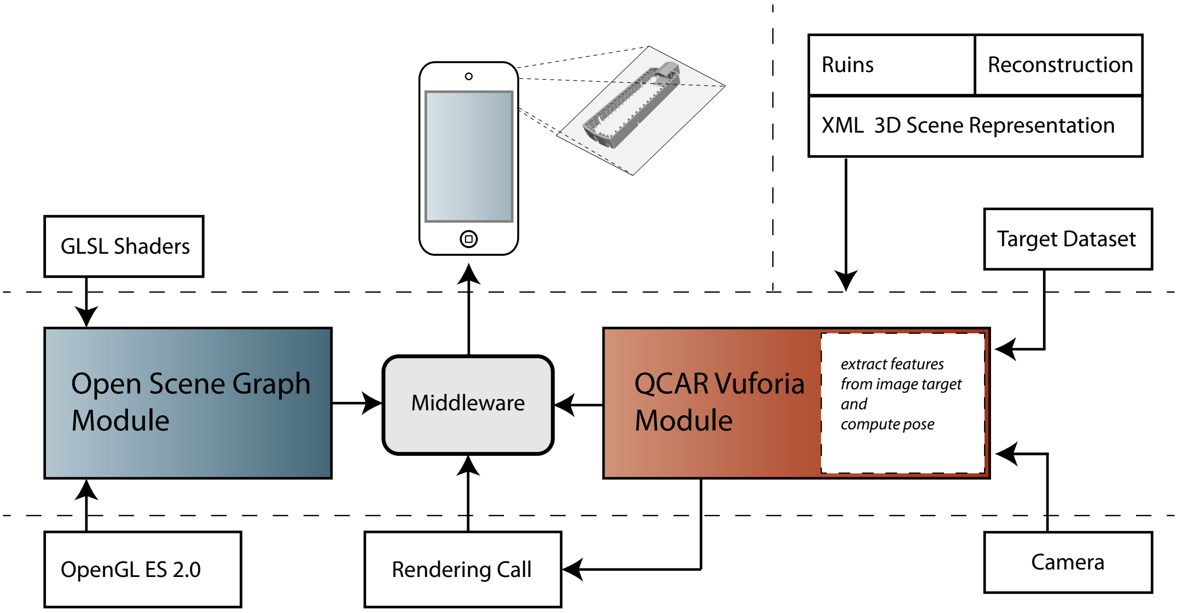

Figure 2: Schematic view of the framework components.

3.1 Vuforia SDK

Vuforia by Qualcomm [5] is a mature platform aimed to a wide range of handheld devices, supporting both iOS and Android; it is released for free, even for commercial purposes. Within our framework, Vuforia is used for the markerless tracking of the camera images. Feature based representations of relative user-defined *image targets* are created for each monument.

To create these targets, the developer should upload the image that needs to be tracked on Qualcomm servers by using a simple web interface, namely the Target Management System. Afterwards the developer can download the relative target resources bundled in a dataset, which can then be loaded at runtime through a simple API call. Such a dataset contains an XML configuration file allowing the developer to configure certain trackables' attributes, and a binary file containing a representation of the trackables.

The accuracy of the tracking is based on the number and the quality of the relevant features extracted by the Target Management System. For achieving a good tracking quality, input images must 1) be rich in detail, 2) have optimal contrast and 3) avoid repetitive patterns like grassy fields, the facade of modern house with identical windows, a checkerboard and other regular grids. Once built, the datasets are loaded during the application initialization. Only one dataset can be active at a certain moment, however a dataset can contain multiple targets.

Our framework handles the following core components of Vuforia:

Camera. The camera component manages the capture of each frame from the device camera, and transmits the data to the tracker. The camera frame is automatically delivered in a device dependent image format and size suitable both for OpenGL ES rendering (see Sec. 3.2) and for tracking.

Tracker. The tracker component implements algorithms for detecting and tracking real world objects from the camera video frames. The results are stored in a state object that is used by the video background renderer and can be accessed from the application code. A single dataset can contain multiple targets. Although the tracker can load multiple datasets, only one can be active at a time. Our system is designed in such a way to scan multiple datasets and perform an automatic dataset switch detection.

Video Background Renderer. The video background renderer component renders the camera image stored in the state object. The performance of the background video rendering is optimized for a wide range of handheld devices.

Our framework initializes the above components and for each processed camera frame, the state object is updated and the render method is called. The framework queries the state object for newly detected targets and updates its logic with the new input data, then it renders the virtual graphics overlay.

As the image targets and the virtual contents (3d models, videos) are loaded in memory, the framework continuosly test for the presence of the image targets in the camera field of view. The test is performed at the beginning of each rendering cycle. When the targets are detected and the pose estimated, proper affine transformations such as translation, rotation and scaling are performed in order to correctly render the 3D model and align it to the real environment.

3.2 OpenSceneGraph

OpenSceneGraph [21] is a robust and mature open-source, high-performance 3D graphics toolkit used for developing interactive applications in many different fields like visual simulation, games, virtual reality, scientific visualization and modelling. It is cross-platform and its functionalities are accessible through portable standard C++ code.

OpenSceneGraph uses a scene graph to represent the spatial and logical relationship of the graphical scene. The scene graph is a hierarchical graph not containing direct cycles and isolated nodes, starting from a root node located at the top level. Each node can contain any number of children; entities stored in a node (e.g., 3D models), share common operations and data.

The actual 3D rendering is performed via the OpenGL ES 2.0 API. Such an interface is cross-platform, open-source and designed for embedded systems like hand-held devices. Since OpenGL ES 2.0 relies on the use of the programmable pipeline, all the effects, lights, materials and the rendering itself have been managed using shaders.

Although all the rendering setup was realized using OpenSceneGraph, the rendering update is called by Vuforia (see Sec. 3.1), so that framerate is kept synchronized with the speed of pose computation. When the rendering does not involve augmenting reality (i.e., when a single model part is shown), the update function is controlled through a timer object handled by the operating system that allows the application for synchronizing its drawing to the refresh rate of the display.

The OpenSceneGraph module is also responsible for the loading of the 3D models. Usually, a model representing a complex architectural monument is composed by several meshes. These meshes are stored as separate files; at loading time, each mesh is loaded and assigned to a node of the scene graph. All the meshes are loaded concurrently using a secondary thread. Most of the meshes are loaded during the initialization phase when the framework is launched, other ones are loaded on-demand according to the user interaction.

3.3 Structure of the system

The framework is built according to the Model-View-Controller (MVC) design pattern [14]. The management of the application is delegated to a main view controller that initializes all the subcomponents: the Commands view controller, the AR view controller and the OSG view controller.

The Commands view controller is the class devoted to manage all the widgets available in the user interface, from the buttons to the virtual object selection. This controller captures every command issued by the user and instantiates the execution of an appropriate action, which is often delegated to the appropriate controller. For example, in the case of virtual object picking the computation of which object has been selected is delegated to the OSG view controller. The AR view controller manages the lower part of the augmented reality system. Its main function is to initialize and set up the modules for augmented reality, load the appropriate data set for the targets, and ultimately call the rendering update function, which is then delegated to the OSG view controller. The OSG view controller manages all the rendering calls using the OpenSceneGraph libraries and is responsible for setting up the shaders, loading the 3D object models in memory and managing 3D animations.

The tracking system relies on marker-less feature tracking and vision algorithms to estimate the pose of some predefined image targets. The information regarding such image targets is incorporated in a dataset. Once the dataset is loaded in memory the application is authorized to search for the corresponding targets in the field of view of the camera. After the pose is extracted the 3D models on the scene graph are enabled or disabled based on the application status, then the shader uniforms are updated with the computed model view and projection matrices. For our application we created a custom target image representing a plan of the Imperial Fora area. Such image is designed to be understandable even without the AR layer and to be printed in information boards near the ruins of the Imperial Fora or in AR cards.

4 USER INTERFACE

Given the manipulative interaction style of augmented reality systems, the traditional WIMP (Windows, Icons, Menu, Pointing) interface is not suited for this framework [17]. Rather, the final user interacts with the augmented view by touching the screen of the handheld device. Whenever the framework detects a touch, a line segment intersector is instantiated in order to precisely pick the corresponding virtual object on the screen. The scene graph tree currently rendered by the camera is scanned for intersections. In particular an intersection visitor is executed on the rendering camera, and if one or more intersections are found, then nodes on the path to the first intersection are visited until we find a node labeled for interaction. The selected node is returned to the commands controller which issues a proper action according to the type of object that user touched. Using this fast ray-casting technique, the framework guarantees the interactivity of the user interface even for models composed by a large amount of meshes.

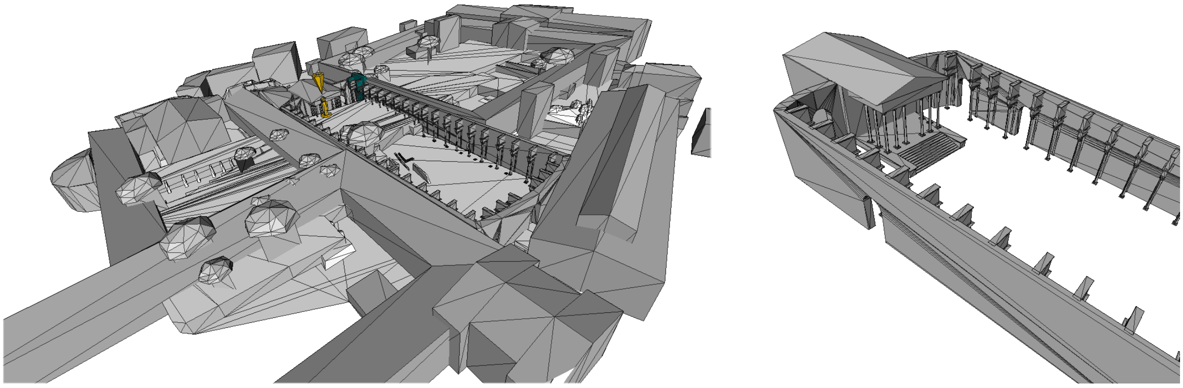

Figure 3: A 3D model representing the Forum of Nerva. It is a set of low-polygonal meshes optimized for interactive rendering and low memory usage.

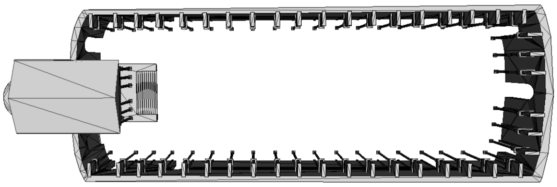

Figure 4: Top view of the 3D representation of the Forum of Nerva.

4.1 3D Visualization

When the user points the camera of the device towards a real area identified by an image target (Sec. 3.1), a 3D virtual model is superimposed to the video feed, providing an alternative representation of the reality. Besides their constant improvement, handheld platforms are in general limited in computational performance and main memory capacity when compared to desktop solutions. Thus, the number of polygons and the related graphics quality detail must be devised in order to be smoothly visualized on a handheld device without sacrificing meaningful details. Fig. 3 and Fig. 4 show the low-polygonal reconstruction of the Forum of Nerva, suitable to be embedded in our system.

For each model, the framework supports the visualization of different historical versions. For example, Fig. 5 shows two different versions of the Forum of Nerva in two different ages, the Imperial Roman age and nowadays.

The 3D model is logically organized in several sensible areas, which can be magnified and interactively manipulated. In this case a new view is presented to the user showing a high-quality rendering of the interested area. The user can zoom in and out the model through a pinch; 3DOFs rotation of the object (roll-pitch-yaw) is performed using an arcball [16] and the scene can be reset to default with a double tap. Fig. 6 shows an example of separate views for details of the Forum of Nerva.

4.2 Virtual Video Streaming

The framework provides the functionality for rendering videos in virtual objects. This is used to create virtual animated buttons and to present informative videos to the user. The videos are visualized beside the 3D model in augmented reality. User can interact with the virtual video in a natural way by tapping for start, pause and resume. Double tapping puts the video in full-screen mode. This has been achieved by exploiting the render-to-texture functionalities of OpenSceneGraph in conjunction with the ffmpeg library, which provides support for many types of video containers and codecs. Fig. 7 shows an example of virtual video applied at the reconstruction of the Forum of Nerva.

5 RESULTS AND CONCLUSIONS

A prototype of the framework has been implemented on top of the iOS operating system; thus it runs on devices such as iPhone, iPad and iPod Touch, that are mainly controlled through the touch interface and are capable of 3D graphics acceleration. However, given the cross-platform nature of all the involved software components, the same framework is easily implementable on other platforms such like Android. The framework achieves interactive rendering of models with over 11000 vertices at 30 frames per second (FPS) on current generation devices mounting a dual core A5 chipset. On previous generation devices (like the iPod Touch 4th generation, single core A4 chipset), the frame rate drops to 5 FPS which can be still considered usable.

The framework allows for the interactive exploration of cultural heritage sites, providing the final users with the possibility to interact with the digital contents in a natural way by using common handheld devices. Through augmented reality techniques, the system provides a solution for the identification of archaeological artifacts and the comparison with their different historical versions.

Our framework allows the visualization of different historical versions of an ancient artifact directly where it

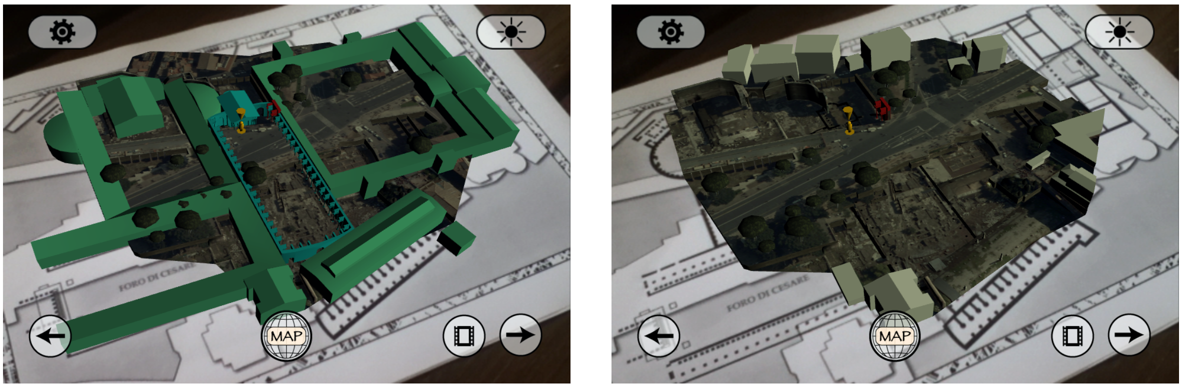

Figure 5: Representation of the Forum of Nerva in (a) the Imperial Roman age and (b) in the current days. Note the red model, namely *Colonnacce*, which is the only remaining part of the forum.

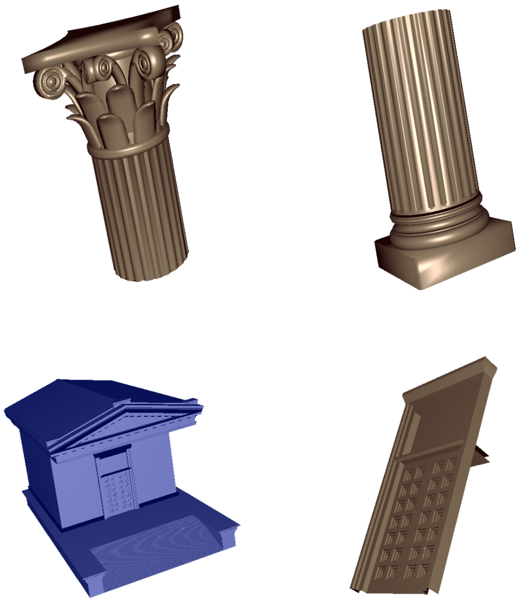

Figure 6: Details of the 3D model can be magnified and explored separately from the augmented view.

was placed originally. The user points the camera of the device towards the on-site ruins' relative target, while the software tracks it and overlays an interactive virtual 3D model of the artifact on it. Some of the most meaningful parts of the model can be selected and magnified to be observed in detail. Special areas of the user interface are devised as 3D *video buttons* embedded into the model. The user can watch the related video together with the 3D model, or in full-screen mode.

The applicability of the framework is tested by providing an augmented view of the Ancient Forum of Nerva. The 3D model has been based and realized according to the documentation provided by historians in the field [18]. The augmented view is superimposed to a sensible map located near the ruins of the ancient forum of Nerva, so that the visitors can observe how the forum appeared during different historical ages. By tapping on meaningful parts of the model, like the so-called *Colonnacce*, user has access to historical information, audio guide, photo gallery and more about the selected monumental artifacts.

After extracting the pose using an image target at a specific fixed position in the environment, it is possible to project the object at the ground level or in a specific relative position with a simple geometric transformation. However this situation constrains the user to observe the virtual environment without loosing the target. Indeed the target loss would cause a lack of information necessary to adjust the relative pose. A possible idea for a future research development may be to compensate this loss of information by estimating camera motion using the data coming from sensors such as GPS, compass and accelerometers, that are usually integrated in handheld devices. In this way user will no more be constrained to maintain the target inside the field of view of the camera.

Plans for future development include further improvement of the user interface and extensive tests on a wider range of datasets (both image targets and 3D representations). Furthermore, we envision this framework to be particularly suitable for the next generation of AR peripherals like the Google Project Glass [2].

6 REFERENCES

[1] i-Mibac Voyager, March 2013.

[2] Project Glass, March 2013.

[3] Rome MVR, March 2013.

[4] Rome View, March 2013.

[5] Vuforia SDK, March 2013.

[6] Ronald Azuma. Tracking requirements for augmented reality. *Commun. ACM*, 36(7):50–51, July 1993.

[7] Ronald Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355 – 385, August 1997.

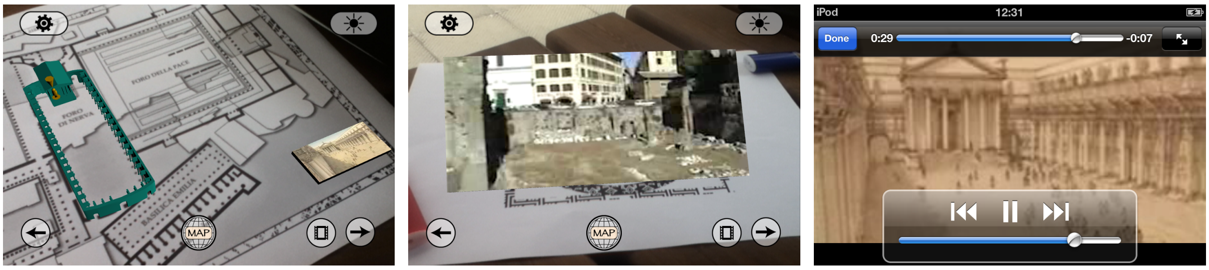

Figure 7: Videos embedded into the augmented view.

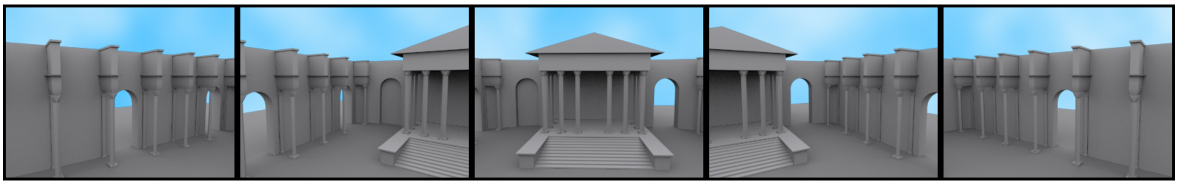

Figure 8: Conceptual rendering example of different camera views, simulating the user turning around inside the Forum of Nerva.

[8] Erich Bruns, Benjamnin Brombach, Thomas Zeidler, and Oliver Bimber. Enabling mobile phones to support large-scale museum guidance. *IEEE MultiMedia*, 14(2):16–25, April 2007.

[9] Julie Carmigniani, Borko Furht, Marco Anisetti, Paolo Ceravolo, Ernesto Damiani, and Misa Ivkovic. Augmented reality technologies, systems and applications. *Multimedia Tools Appl.*, 51(1):341–377, January 2011.

[10] Omar Choudary, Vincent Charvillat, Romulus Grigoras, and Pierre Gurdjos. March: mobile augmented reality for cultural heritage. In *Proceedings of the 17th ACM international conference on Multimedia*, MM '09, pages 1023–1024, New York, NY, USA, 2009. ACM.

[11] P. Dähne and J. Karigiannis. Archeoguide: System architecture of a mobile outdoor augmented reality system. *Mixed and Augmented Reality, IEEE / ACM International Symposium on*, 0:263, 2002.

[12] Jiang Gao. Hybrid tracking and visual search. In *Proceedings of the 16th ACM international conference on Multimedia*, MM '08, pages 909–912, New York, NY, USA, 2008. ACM.

[13] Tim Gleue and Patrick Dähne. Design and implementation of a mobile device for outdoor augmented reality in the archeoguide project. In *Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, VAST '01, pages 161–168, New York, NY, USA, 2001. ACM.

[14] Glenn E. Krasner and Stephen T. Pope. A cookbook for using the model-view controller user interface paradigm in smalltalk-80. *J. Object Oriented Program.*, 1(3):26–49, August 1988.

[15] George Papagiannakis, Sébastien Schertenleib, Brian O'Kennedy, Marlene Arevalo-Poizat, Nadia Magnenat-Thalmann, Andrew Stoddart, and Daniel Thalmann. Mixing virtual and real scenes in the site of ancient pompeii: Research articles. *Comput. Animat. Virtual Worlds*, 16(1):11–24, February 2005.

[16] Ken Shoemake. Graphics gems iv. chapter Arcball rotation control, pages 175–192. Academic Press Professional, Inc., San Diego, CA, USA, 1994.

[17] Andries van Dam. Post-wimp user interfaces. *Commun. ACM*, 40(2):63–67, February 1997.

[18] A. Viscogliosi. Il foro transitiorio. *Divus Vespasianus. Il bimillenario dei Flavi.*, pages 202–208, 2010.

[19] Vassilios Vlahakis, Nikolaos Ioannidis, John Karigiannis, Manolis Tsotros, Michael Gounaris, Didier Stricker, Tim Gleue, Patrick Daehne, and Luís Almeida. Archeoguide: An augmented reality guide for archaeological sites. *IEEE Comput. Graph. Appl.*, 22(5):52–60, September 2002.

[20] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose tracking from natural features on mobile phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '08, pages 125–134, Washington, DC, USA, 2008. IEEE Computer Society.

[21] R. Wang and X. Qian. *Openscenegraph 3.0: Beginner's Guide*. Packt open source. PACKT PUB, 2010.