Augmented Visualization on Handheld Devices for Cultural Heritage

Giovanni Murru
Sapienza Univ. of Rome
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 currently explored 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 and immerse 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 which 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 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 accessible to the mass market.

1.1 Contributions

In this paper, we present a novel mobile framework able to run on a modern handheld device. 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 smartphone 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, a part of the Imperial forums in the Roman Empire age. The 3D model has been built according to the information acquired from previous archaeological studies [18].
2 RELATED WORK

Augmented Reality is a technology allowing for extending the vision of the real world with the 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 enhance it with digital data, making virtual and real objects coexisting 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 the position and the 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 display 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 proximity, 3-axis gyroscope, ambient light, accelerometers, magnetometer and assisted GPS and are hence 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 replace a 3D model with
the video feed coming from the camera, requiring the user to stand in a known position and implementing 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 based on feature detection and those using a model of the tracked object’s features.

The framework employs the markerless, video tracking capabilities of the Vuforia SDK [5] to calculate in real time the real camera position and orientation (pose) relative to some predefined image targets. Once the pose is retrieved, a virtual camera is rendered using the OpenSceneGraph module [21], in order to properly overlay the 3D computer graphics model onto the markers.

Hence, the proposed framework is based on two main software components, Vuforia and OpenSceneGraph, and the interfaces towards the user (see Fig. 2). Vuforia is the module responsible for tracking an image target and estimate its pose in 3D space, while OpenSceneGraph takes care of the rendering of the models and user interfaces. The most important features of these packages, and how they are employed within the framework, are briefly depicted in the following sections.

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. For each monument, some image targets are created, each of them contains information on which features to search during the tracking phase.

To create these target, the developer uploads an image for the trackables on Qualcomm servers using a simple web interface, namely the Target Management System, and then downloads the relative target resources bundled in a dataset, which can be then loaded in the framework through a simple API call. Such a dataset contain an XML configuration file allowing the developer to configure certain trackables features, and a binary file containing the trackables database. These assets are bundled in the framework and used at run-time.

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 continuously 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 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 handheld devices. Since OpenGL ES 2.0 relies on the use of the programmable pipeline, all the effects, lights, materials and the rendering itself have to be managed using shaders.

Although all the rendering setup was realized using OpenSceneGraph, the rendering update is called through Vuforia (see Sec. 3.1), which controls the visualization flow of the entire application. When the visualization does not involve augmented views (e.g., when a single model part is shown), the render update function is delegated to a timer object handled by the operative 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 serialized 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]. There are three main controllers: the Commands view controller, the AR view controller and the OSG view controller.

The Commands view controller is the class demanded 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. 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 control the AR EAGL view, a view used to display and control the lower part of the rendering system as the call to the rendering update function, which is 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.

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 instanced in order to precisely pick the corresponding virtual object on the screen. The point touched on the screen is projected on the near and on the far plane of the frustum view, providing a line which span the 3D scene. Then, the whole scene graph tree is scanned and each node is tested for intersection with the line; for every node with more than one child, the branch not intersected by the line is pruned and not considered for further test. 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. The node selected by the user is returned to the commands controller which issues a proper action according to the type of the object.

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 computation performance and memory storing capacity when compared to desktop solutions. Thus, the number of polygons and the related 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 framework.

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
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 allow for 3D 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, A4 chipset), the frame rate drops to 5 FPS which can be still considered interactive.

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 provide a solution for the identification of archaeological artifacts and the comparison with its different historical versions.

Our framework allows for visualizing different historical versions of an ancient artifacts directly were it was placed originally. The user points the camera of the device towards the on-site ruins, the software tracks the video feed and superimpose an interactive virtual 3D model of the artifact. 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.
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.

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. The user can also select meaningful parts of the model, like to so-called *Colonnacce*, by tapping the touch screen, and access to a more detailed view of them. Furthermore, the frameworks allows to add contextual videos and immersive exploration to the augmented view of an archeological artifact.

Plans for future development include further improvement of the user interfaces and extensive tests on the 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

Figure 7: Videos embedded into the augmented view.

Figure 8: Example sequence of different immersive views taken while the user turns around (refer to the attached video).
