LecceAR: An Augmented Reality App

Banterle F.[1], Cardillo F.A. [1,2], Malomo L. [1], Pingi P. [1],
Gabellone F. [2], AmatoG. [1], Scopigno R[1].

1 Istituto di Scienza e Tecnologie dell'Informazione, CNR, Pisa, Italy
2 Istituto per i Beni Archeologici e Monumentali, CNR, Lecce, Italy
{francesco.banterle|franco.alberto.cardillo|luigi.malomo|
paolo.pingi|giuseppe.amato|roberto.scopigno}@isti.cnr.it
francesco.gabellone@ibam.cnr.it

Abstract. This paper discusses a case study on the use of augmented reality (AR) within the context of cultural heritage. We implemented an iOS app for markerless AR that will be exhibited at the MUST museum in Lecce, Italy. The app shows a rich 3D reconstruction of the Roman amphitheater, which is nowadays only partially visible. The use of state-of-the-art algorithms in computer graphics and computer vision allows the viewing and the exploration of the ancient theater in real-time.

Keywords: augmented reality, image matching, image tracking, 3D rendering, mobile app.

1 Introduction

Azuma [1] defines the class of Augmented Reality (AR) as those systems that fulfill the following three characteristics:

- combine real and virtual imagery;
- support interactive visualization/manipulation in real-time;
- provide visual content registered in the 3D space.

The key aspect of any AR application is thus the *augmentation* of the physical world with digital information that is aligned with the place/context where the experience takes place. Mobile devices are nowadays equipped with high quality camera sensors and quite powerful processing capabilities (i.e. fast CPUs and GPUs) which have sparked the development of interactive augmented/virtual reality applications for Cultural Heritage (CH). Even though mobile computational power is only a fraction of that offered by current desktop platforms, the widespread diffusion of mobile devices opens up new opportunities for CH operators. They can now provide high quality digital content for engaging visitors with an entertaining AR experience that was not possible a few years ago.

In this paper, we present a mobile app, LecceAR, designed to run on smartphones and tablets, able to overlay a detailed 3D reconstruction on top of a target image. The image is used to select the specific artwork (in the example shown in this paper, the Roman Amphitheater in Lecce) as well as to drive the interaction with

the digital representation. The app, which will be part of the opening exhibition of the Living Lab[1] in the MUST museum in Lecce, Italy, will provide visitors with a unique experience where lost or hidden details of the amphitheater are visible in AR. Since only scarce physical remains of the amphitheater are available at our times (see Fig.Figure 1 left image), rendering a digital reconstruction is the only way to convey its ancient magnificence to the tourists. The Living Lab of the MUST museum is meant to be a physical place where museum visitors can experience and understand heritage by means of ICT tools.

The rest of the paper is organized as follows. The next section provides a short overview of the previous works on AR running on mobile platforms. Section 3 describes briefly the Roman amphitheater and the techniques used to create the digital 3D model. Section 4 describes the app that has been developed for the iOS platform. This is a markerless AR app whose target image, prepared to be aesthetically suited for a museum exhibition, causes several problems in standard matching and tracking algorithms, as discussed in section 0. A strong feature of the app is the capability of rendering high quality and highly complex 3D models on mobile devices, as shown in section 4.2. The final section outlines our plans to further improve and enrich the user experience offered by AR apps, in particular by exploiting the monuments in Lecce and the synergy stemming from the Living Lab initiative.

2 Related work

Augmented Reality has been an active research field for years with a very vast literature, so we restrict ourselves mainly to AR experiences and tools designed for CH. In this section, we briefly discuss the most recent approaches that highlight both the major research topics and the main applications.

Wojciechowski et al. [2] proposed a system, ARCO, which allows the museum staff to build easily augmented and virtual exhibitions using AR and VR technologies.

Ridel et al. [3] presented a system for improving the readability of artifact details which are difficult to distinguish due to deterioration. In order to achieve this, the 3D model is analyzed to capture convexities and concavities. Then, the system projects onto the real object an expressive 3D visualization highlighting its features at interactive time. An electromagnetically tracked prop and optical tracking of the user's fingers are employed to allow interaction with the virtual objects.

Lu et al. [4] used AR for displaying animated paintings on mobile devices. Through statistics collected by the app they were also able to assess how AR technologies can affect the learning or observation of art in a museum.

Miyashita et al. [5] proposed a digital guide, which is based on AR and developed with two main goals: to provide visitors information to enable them to fully understand a piece of art, and to help them to navigate into the exhibition space in a meaningful path.

[1] http://livinglabs.regione.puglia.it/en/home

Systems for AR in outdoors for cultural heritage based on markerless technologies have been also presented [6, 7]. These systems can recognize a real-world cultural heritage site, and then they can place the 3D reconstruction of the site (how it looked in the past) inside the real visual scene.

3 The Roman Amphitheatre in Lecce

In the 15th century A.D., a highborn jurist of Lecce, Iacopo Antonio Ferrari, describing Lecce's Roman amphitheater in his work "Paradossica Archeologica", wrote: "Lecce shows, within its body and in its very Piazza, the most beautiful relics of its antiquity". In 1896, De Giorgi identified several pillars of the amphitheater and discovered the cornices of two arches. From 1901 to 1910, he succeeded in accessing to the external ambulatory, its basement, and even the surrounding cuniculus. This led to an enthusiastic period of investigation, which unearthed parts of the cavea and a huge quantity of architectural and decorative fragments. From 1938 to 1940, the Soprintendenza ai Monumenti e alle Antichità della Puglia (Authority for the Monuments and Antiquities of Apulia) undertook the excavation and restoration works which had led to the current appearance of the amphitheater. The dating of the monument has been a matter of debate for a long time. Many evidences and documents now point to the Augustan period.

Today, only a third of the ancient building is unearthed and visible. However, we can understand from this portion its monumentality and importance during Roman period, around 2,000 years ago.

The amphitheater was designed and built in four distinct sections, punctuated by four entrances, and organized symmetrically along the main axes. The tall building is based on modular repetitions of three grouped arches, regularly appearing along the elliptical perimeter of the arena. This provided, respectively, access via stairway to the upper ambulatory, the lower ambulatory, and to an enclosed space with level paving.

The modular design reappears in the stairways distribution for connecting the lower ambulatory and ima càvea, upper ambulatory and media càvea, perimetric portico and summa càvea, arranged to encourage spectators movement in a double series of oblique paths.

The conglomerate structures feature reticulated facades in blocks of Lecce stone, while the core is a mixture. This includes fat lime mortar, sand and imported Pozzolana, and flakes of Lecce stone. Pumice and volcanic slag, which were certainly imported, were used for all vaults, which were barrel shaped. The building measured 102 by 83 meters externally, with an arena of 53 by 54 meters, and could hold between 12,000 and 14,000 people.

Today the amphitheater of Lecce is partially unearthed. However, the original appearance of the ancient building can be determined with close approximation by studying its archaeological remains. In order to provide a satisfactory 3D reconstruction of the building, we studied the available archaeological and historical data. Moreover, we performed a study on the existing structures: we carried out an indirect survey

Figure 1 Left: Photography of the amphitheater in its current state. Right: Target image used in the AR app. This is the image whose recognition triggers the rendering of the 3D model: the 3D model is kept aligned with the position of the target image in the video frames.

integrating laser scanning and digital photogrammetry technologies. The merging of the data gathered by the two acquisition systems identified morphological and textural elements (related to the color of the surface). We made some plausible assumptions about the original building based on such evidence and according to statics and to the construction techniques in use in the Roman period. The 3D survey was crucial to understand the space distribution inside the building and to verify elevations. This led to results similar to previous studies conducted by researchers at the University of Lecce. From this study and data, the 3D reconstruction was modeled using polygonal and NURBS techniques. This choice allows a lot of flexibility, especially when polishing fine details, and it improves the management of the output 3D model (i.e. the total number of triangles). This is extremely important when targeting mobile devices. The sculptural elements and the architectural details are the result of a digital restoration based on initial models extracted by photomodeling (i.e. 3D reconstruction from photographs).

4 The app LecceAR

LecceAR is an app whose final version will be able to augment points of interest in Lecce by overlaying historical and architectural information over the video stream coming from the camera of the mobile device. It can work with physical target images as well as images gathered from the real world. When a target image is viewed through the camera of a mobile device, the rendering of a high quality 3D model, precisely aligned with the target image, is triggered. The current implementation of LecceAR is designed to recognize and track a specific target image, shown in Fig. 1 (right image). When the target is detected in the video stream acquired by the camera, LecceAR displays the 3D model of the amphitheater, keeping it aligned with the target as the device (or the target) moves. The app does not need to use the specific tar-

get used in the current use case: the target image can be chosen to be any synthetic or natural image. Furthermore, LecceAR can be extended to recognize more than one target image, displaying different digital information (not necessarily 3D models) based on the recognized target.

LecceAR has two main modules:

- the matching and tracking module, which estimates the location and orientation of the mobile device with respect to the target image;
- the 3D rendering module, which visualizes a 3D model, aligning it with the video stream captured by the camera device.

More specifically, the matching and tracking module is responsible of establishing whether or not a target image or a known monument is in the field of view (FOV) of the camera and, if present, tracking it in real-time when the device is moved. This module also continuously computes the correct point of view of the mobile device, with respect to the target image. Based on the input received by the first module, the 3D rendering module computes the transformation needed to display in real-time a 3D model, aligned with the target image.

4.1 The Matching and Tracking Module

The matching and tracking module has first to process frames coming from the camera in order to establish whether or not they contain a known target image. If a frame contains a target, the module computes a geometric transformation mapping the target onto the video frame, and initializes the tracker. Besides the choice of the specific matching and tracking algorithms, the quality of the estimated alignment depends on the specific target images which are to be detected.

Many AR apps recognize only fiducial markers, which are synthetic images composed by black and white dots. When they are not occluded, they increase the detection rate and speed up the computation of the user's pose. Since our app will be part of a museum exhibition, there are "aesthetics" constraints on the target image that will be printed on a 40cm x 40cm plate on top of a stand. The target used in the current implementation, shown in right image of Fig.Figure 1, is an overlay of two pictures: three quarters of the target contains a textured top view of the reconstructed 3D model of the amphitheater, while the upper left quarter contains a symbolic 2D map of the same amphitheater.

The search of the target image in the live video stream is performed using a classical approach based on the following three steps:

- extraction of keypoints and related descriptors;
- comparison between the two sets of descriptors, extracted from the live frame and from the target image, resulting in a set of matches *M;*
- geometric verification of the matches in *M* in order to establish whether or not the target is present in the video frame.

Keypoint extraction and matching. There are several keypoint extraction algorithms for scale and rotation-invariant object recognition with a varying computational cost associated both to the extraction of the visual features and to the comparison among descriptors. For mobile applications, the choice usually falls on binary descriptors because of the low computational cost of the matching stage, based on the Hamming distance. We chose the ORB features (Oriented Fast and Rotated Brief) [8], which provide a good trade-off between quality and real-time performance. According to [8], the L2 distance between two binary ORB descriptors that guarantees a good number of true positive matches is around 64. When using such threshold on the specific target, the number of matches to be verified in the subsequent step is too high and the real-time performance of the app severely reduced. For this reason we decided to use a lower threshold with a value set to 40.

Geometric verification (target recognition). Once obtained a set of matches between the two sets of keypoints extracted from the known target and the video frame, we need to verify that the relative positions of the matching keypoints are compatible with a specific geometric transformation of the target image. This is accomplished using the following steps:

- Use RANSAC [9] to determine a homography H mapping the target image onto the frame (query image). Basically, four random points are selected in order to compute the parameter of H. The homography is then applied to all the matching couples of keypoints for counting the number of matches compatible with H (inliers). This process is repeated K times: the RANSAC returns the matrix H with the largest number on inliers.
- Apply the inverse homography H' to the query image and perform a second iteration of RANSAC to establish a second homography $H2$ between the target T and the video frame. If a second homography is found, then use the refined homography, otherwise fail and start over.
- Once a good homography is found, start both the tracking and the rendering module.

Sometimes the RANSAC algorithm might return a degenerate homography. In order to avoid such false matches, we apply some heuristics [10] able to discard homographies that are likely to be wrong. The first heuristics concerns the determinant of H, $det(H)$. Since H is invertible, when $det(H)$ is very close to zero, it is likely that H is degenerate. However, if the determinant is too large, H', whose determinant is 1/det(H), is almost singular. Based on the previous consideration we only accept a homography H if its determinant is in the range $[k, 1/k]$ with $k=0.1$. Some homographies, which pass the previous test, do not preserve the convexity: they transform the target image, which is a square, into a concave quadrilateral. In order to determine the convexity of the target mapped by the homography we added a check on the sign of the cross product of all the adjacent edges: if all the cross products have the same sign then the polygon is convex. Finally, we check the area of the projection; if it is too small the homography is rejected.

Tracking. Once the homography localizing the target in the video frame is available, the tracking is initialized. The tracked features are the points selected by the algorithm `Good Features To Track` [12]. The tracking starts only if there are more than 30 good points (hereafter called tracks). The locations of the points are passed to a procedure implementing a Lucas-Kanade optical flow computation based on a pyramid of images [11]. Once we have the locations of the tracks at time t+1, we re-compute a homography using the same algorithms as the Matching module. The global geometric transformation, localizing the target in the new video frame, is obtained by composing the homography estimated at time t with the one estimated at time t+1.

Some tracks are normally lost between two consecutive frames. If we used only the initial set of tracks, we would be forced to recalculate a homography after few iterations of the tracking algorithm. Unfortunately, the target image can be recognized only if the upper left sector is in the FOV of the camera: the rest of the image contains keypoints that, basically, match each other and prevent the algorithm from reaching a geometrically validated match. When the tracking fails, the app would not be able to restart the tracking if the user is viewing the target area containing only the textured top-view of the model,. In order to keep tracking as long as possible, we extract again the points to track if at iteration *t* the points preserved are less than a threshold (1.5 the initial number of tracks, which corresponds to a number that allows a reliable estimation of the homography).

4.2 The Rendering Module

Once a homography *H*, is computed, a camera pose is computed from it in order to render synthetic 3D models using the OpenGL library [13]. From, *H*, the rotation matrix, *R*, of the device is extracted from its first two columns; the third column is computed as cross product between these twos. Furthermore, the last column of *H* encodes the position of the mobile device.

Note that before extracting the camera pose, *H* has to be multiplied by the inverse of the intrinsics matrix, *K*, in order to convert the camera pose from the image space to the world-space. In our case, *K* has been previously calibrated using the Camera Calibration Toolbox for MATLAB[2].

At this point, the camera pose is fully recovered, i.e. the rotation matrix and the translation vector, and only the scene scale factor, *s*, is missing. This is computed by dividing the original size of the target image by the size of the target in the video stream in pixels in a straightforward way. Finally, the camera pose, *s*, and *K* are used to set-up a classic OpenGL matrix for rendering virtual 3D models.

Although our tracker outputs a stable homography matrix, a small amount of jitter (i.e. camera shaking) can be still perceived in the computed pose. Therefore, in order

[2] "Camera Calibration Toolbox for MATALB". Jean-Yves Bouguet. http://www.vision.caltech.edu/bouguetj/calib_doc/ (accessed in July 2015)

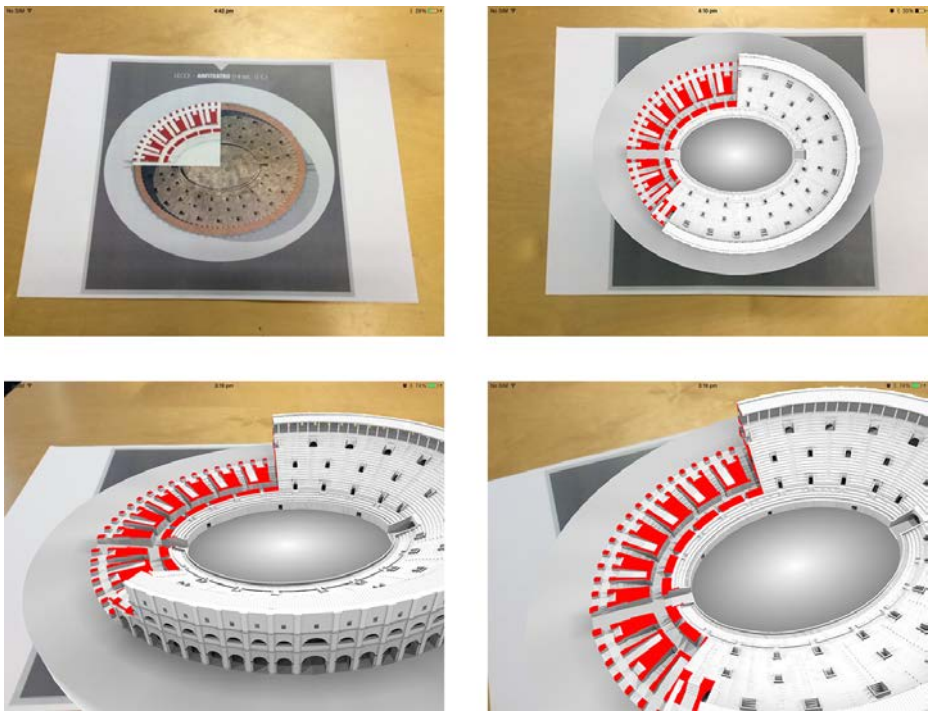

Figure 2 Screenshots of the app during an actual session. From the upper left image, clockwise: the target as framed by the device, the 3D model shown in overlay on the camera live feed, the rendering when the user moves the device closer to the target.

to have a smoother camera motion, we implemented a linear Kalman filter only for the translation vector, *t*, of the pose. We noticed that a linear Kalman filter applied also to the rotation matrix, *R*, encoded as Euler angles, works but it suffers from the gimbal lock. To avoid this issue while keeping a simple solution, we implemented an alpha-beta filter for the rotation matrix, encoded as a quaternion, using a spherical linear interpolation.

Renderer. We also developed a proprietary component for rendering virtual objects, called Viewer3D. This is an OpenGL|ES 2.0 real-time renderer which was developed for the iOS platform using the VCG library[3]. The renderer, whose setup requires only a few lines of code, can be encapsulated inside a UIView. In this way, the renderer is very handy, because a UIView is the basic iOS widget for visualizing graphics on a screen; it can be conceived as a window in desktop-system terminology.

Once the renderer is initialized, 3D models can be rendered by loading them and assigning a shader. Viewer3D supports different file formats, including the PLY and OBJ formats, that are the de-facto standards for 3D scanned models and 3D modeling packages used in the CH domain. Moreover, our renderer allows to define dif-

[3] http://vcg.sourceforge.net/

ferent attributes over the vertices of a 3D model, to encode normals, texture coordinates, colors, ambient occlusion, etc. This increases the flexibility of the system in coping with different rendering and shading needs.
To better support AR applications, we added a transparency parameter which fades in and out the visualization of the 3D models. In particular, the fade-out effect gracefully hides the 3D models when the tracking of the target image fails.

Concerning the performance, our renderer can achieve up to 60 fps while rendering a 3D model composed of 2M triangles, with a texture and using a Phong lighting shader on an iPad Air 1st generation and an iPhone 5S. This is achieved without the need of streaming from the flash memory. Fig. 2 shows some screenshots recorded during an actual session of the app running on an iPad. As can be seen in the picture, the interaction between the tracker and the rendering modules allows the app to align correctly the 3D model in the space where the experience takes place.

5 Conclusions

We have presented LecceAR, a markerless AR app for the iOS platform. This app can recognize and track a target image while rendering a complex 3D model on top of the target image.

We plan to extend our work in different directions. Firstly, we are planning to add an index structure for efficient similarity search, which will enable the app to recognize and track thousands of images. Secondly, we are investigating on efficient and effective techniques to track multiple targets. These allow to point to several target images simultaneously and to provide users with multiple information. Finally, we would like to extend the system to work on non-planar scenes, i.e using directly the 3D metric of the physical world.

Acknowledgements

This work has been partially supported by the Italian research project “DiCeT: Living Lab di Cultura e Tecnologia” (PON04a2_D).

References

1. Azuma, R.T.: A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments. 6 (4), 355-385 (1997)
2. Wojciechowski, R., et al.: Building virtual and augmented reality museum exhibitions. In: Proceedings of the ninth international conference on 3D Web technology. ACM, (2004).
3. Ridel, B., et al.: The Revealing Flashlight: Interactive spatial augmented reality for detail exploration of cultural heritage artifacts. Journal on Computing and Cultural Heritage (JOCCH) 7(2), 6 (2014)
4. Lu, W., et al: Effects of mobile AR-enabled interactions on retention and transfer for learning in art museum contexts. In: IEEE International Symposium on Mixed and

Augmented Reality-Media, Art, Social Science, Humanities and Design (ISMAR-MASH'D) (2014).
5. Miyashita, T., et al: An augmented reality museum guide. In: Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality. IEEE Computer Society, (2008)
6. Han, J.-G., et al: Cultural Heritage Sites Visualization System based on Outdoor Augmented Reality. AASRI Conference on Intelligent Systems and Control, pp. 64-71 (2013)
7. Verykokou, S., Charalabos, I., Kontogianni, G.: 3D Visualization via Augmented Reality: The Case of the Middle Stoa in the Ancient Agora of Athens. Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection. Springer International Publishing, pp. 279-289 (2014)
8. Rublee, E., et al.: ORB: an efficient alternative to SIFT or SURF. In: *Proc. Of IEEE International Conference on Computer Vision (ICCV), (*2011)
9. Fischler, M.A., Bolles R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6), pp. 381-395, (1981)
10. Kumar, D.S., Jawahar, C. V.: Robust homography-based control for camera positioning in piecewise planar environments. *Computer Vision, Graphics and Image Processing*. Springer Berlin Heidelberg, pp. 906-918, (2006)
11. Bouguet, J.-Y.: Pyramidal implementation of the affine Lucas Kanade feature tracker: description of the algorithm." *Intel Corporation* 5 (2001)
12. Tomasi, C., Shi, J.: Good features to track. In: Proceedings of the Computer Vision and Pattern Recognition Conference *CVPR94*, pp. 593-600 (1994)
13. "OpenGL SuperBible. 6th Edition". Graham Sellers, Richard S. Jr. Wright, Nicholas Haemel. Addison Wesley, July 2013. ISBN 978-0321902948
14. Simon, G., Fitzgibbon, A.W., Zisserman A.: Markerless tracking using planar structures in the scene. In: Proceedings Of the IEEE and ACM International Symposium on Augmented Reality, *(ISAR 2000)*.. IEEE, (2000)