A PATTERN-BASED AUGMENTED REALITY APPLICATION FOR THE DISSEMINATION OF CULTURAL HERITAGE

A.-M. Boutsi*, S. Verykokou, S. Soile, C. Ioannidis

Laboratory of Photogrammetry, School of Rural, Surveying and Geoinformatics Engineering, NTUA, Greece

KEY WORDS: Cultural Heritage, Augmented Reality, Computer Vision, mobile app, Photogrammetry, 3D modelling

ABSTRACT:

Augmented Reality (AR) is more than an added value for Cultural Heritage (CH); it is vital for its sustainability, promotion and dissemination, increasing accessibility in CH even during difficult periods of time, like the Covid-19 pandemic. In order to be meaningful and engaging, an AR application should have the following characteristics: easiness of use, high-quality representations and compatibility. This paper presents a marker-less mobile AR application for the display and inspection of high-resolution 3D cultural assets, overlayed on a particular location in the real-world scene. Instead of predefined markers, an image captured by the user is exploited as a pattern for real-time feature matching, pose estimation and scene augmentation. Our approach is based on pure computer vision and photogrammetric techniques, implemented using native C++ and Java code for Android mobile platforms. It is built with the use of the OpenCV library and the OpenGL ES graphics API without any dependencies of AR Software Development Kits (SDKs). Therefore, it supports cross-vendor portability regarding mobile model devices and hardware specifications. The evaluation of the developed application examines the performance of various matching techniques and the overall responsiveness of processing and 3D rendering on mid-range and low-end smartphones. The results showcase the reliability and responsiveness of the pattern recognition as well as the potential of the 3D graphics engine to render and overlay complex 3D models balancing between visual quality and time. The proposed methodology is applied to the Ciborium of the church of St. Charalabos, located at St. Stephen's Monastery in Meteora, Greece.

1. INTRODUCTION

Immersive technologies provide the opportunity of rethinking and further strengthening the role of Cultural Heritage (CH) in the new digital era. Unlike Virtual Reality (VR), the role of Augmented Reality (AR) is not to replace the real objects; it is to encounter and perceive them either from a cognitive or emotional point of view. It allows for completing, precising and reconstructing layers of information on the physical sites or assets through the camera. During the last twenty years, it has offered very interesting possibilities for the promotion of CH through interactive display of heritage items. One of the first AR systems targeted to CH was Archeoguide (Vlahakis et al., 2002), which permitted visualization of 3D monuments of the archaeological site of Olympia, Greece as they were in antiquity. Several AR projects related to CH were reported in the years that followed, e.g., LIFEPLUS (Papagiannakis et al., 2002) and iTacitus (Zoellner et al., 2007), and applications for museums have also been implemented (e.g., Caarls et al., 2009; Vanoni et al., 2012; Venigalla and Chimalakonda, 2019; Khan et al., 2021).

The most widespread and easy to use type of AR technology is Mobile Augmented Reality (MAR) for handheld computers like smartphones and tablets (Siriwardhana et al., 2021). Proof of its wide scale adoption is the abundance of CH AR applications for mobile devices that have been launched (e.g., Haugstvedt and Krogstie, 2012; Verykokou et al., 2014; Galatis et al., 2016; Panou et al., 2018; Ramtohul and Khedo, 2019; Čejka et al., 2020). The nature of this technology can also serve the actual needs of the CH community with the overlay of specific-domain knowledge or by addressing critical situations. Portability along with remote and safe interaction tackles the lack of accessibility for risk management, vulnerability assessment or other issues imposed by natural or human-caused disasters (Zhu & Li, 2021). MAR technology for the dissemination of CH without requiring physical contact with the heritage item has been especially investigated since 2020 (Kunjir and Patil, 2020; Trunfio et al., 2020), because of the social distancing recommended by medical experts due to the Covid-19 pandemic. In that sense, MAR is pervasive in location and applicability of use. Its convergence with photogrammetry for the augmentation of realistic and geometrically precise 3D reconstructions further evolves its potential.

In this paper an open-source Android application for pattern-based AR in the CH section is presented. Real-world surfaces around the end-user can be used as a reference for enabling location-specific AR experiences. The application is capable of real-time processing of captured images for the visualization of high-resolution 3D models as AR overlays. The implementation is based on low and mid-level programming with Java and C++ and it may run on any device with Android operating system and a camera sensor. Unlike the majority of similar AR applications, no AR SDK or other specialized software is used. Each stage of computing and rendering process is described explicitly based on the OpenCV Computer Vision library (OpenCV Team, 2021) and the OpenGL ES graphics API (Khronos Group, 2021). Thus, this work is technologically innovative in terms of hardware independence, visual quality and performance. It integrates the following fundamental concepts:

- feature extraction and matching;
- anchoring based on keypoints and their descriptors;
- high performance 3D rendering and visualization.

The proposed methodology is applied at a characteristic example of inaccessible for the wide audience cultural asset, the Ciborium of the church of St. Charalabos of St. Stephen's Monastery in Meteora, Greece. It is a religious architectural structure of

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-VIII-M-1-2021-57-2021 | © Author(s) 2021. CC BY 4.0 License.

Eastern Orthodox churches, located at the altar of the church which is by tradition, a restricted area to the priest. The 3D model of the Ciborium that derives from image-based photogrammetry and surveying documentation, constitutes the overlay that is promoted and disseminated in the context of this application. The attained performance stability and visual quality as well as the compatibility with any computing device with a camera sensor demonstrate the potential of the application to serve a variety of AR cases in the CH field.

The rest of the paper is organized as follows. Section 2 describes the methodological approach adopted by the AR application using photogrammetric and computer vision techniques for the augmentation of each camera frame. Section 3 presents the application, outlining implementation details, information on the generation of the 3D model of the case study and the achieved results. Finally, the conclusions of this research along with insights on future research steps are discussed in section 4.

2. METHODOLOGICAL APPROACH

In this section, the photogrammetric – computer vision methodology along with an overview of the basic principles of computer graphics followed by the AR application are presented.

2.1 Pattern Object Definition

The pattern object is assumed to be a planar rectangle surface. A nadir image of the pattern object is captured by the user, being a prerequisite for augmentation of the real-world scene. The origin of the world coordinate system is located at the center of the pattern object. *X* and *Y* axes lie on the pattern object plane, while *Z* axis is perpendicular to it, pointing to the camera. The *X* and *Y* coordinates of the four corners of the pattern object are derived from the normalized width and height of the pattern image (equation (1)), ranging from -1 to 1. *Z* coordinate is set to zero.

$$w_n = \frac{cols}{\max(rows, cols)}, \qquad h_n = \frac{rows}{\max(rows, cols)} \tag{1}$$

where w_n = normalized width of the pattern image
h_n = normalized height of the pattern image
cols = number of columns of the pattern image
rows = number of rows of the pattern image

2.2 Feature Extraction and Image Matching

Feature points are extracted once in the pattern image and in every camera frame, using the ORB (Oriented FAST and Rotated BRIEF) detector (Rublee et al., 2011), as it is computationally more efficient than robust well-established detectors like SIFT (Lowe, 2004) and SURF (Bay et al., 2008) and has been claimed to have similar matching performance with them. ORB builds on FAST keypoint detector (Rosten and Drummond 2006) and BRIEF descriptor (Calonder et al., 2010), being rotation invariant and resistant to noise.

During the image matching stage, which is executed based on the minimum Hamming distance between the descriptors of the extracted feature points, outliers are rejected via the RANSAC algorithm (Fischler and Bolles, 1981), through computation of the 2D homography (projective transformation) between the image of the pattern object in each camera frame and the pattern image.

2.3 Pattern Recognition

The correspondences that verify the homography computed by RANSAC constitute the inliers. If a minimum number of inliers is detected, pattern recognition takes place and the 3D model is rendered, superimposed on the camera frame. Otherwise, the scene is not augmented, as the pattern object cannot be recognized in the frame. The initial estimation of the homography obtained by RANSAC is refined using all the inliers, via the Levenberg-Marquardt non-linear optimization algorithm (Moré, 1978). The recognition of the pattern object in each camera frame is accomplished through calculation of the pixel coordinates of its four corners, using the estimated homography matrix and the pixel coordinates of the four corners of the pattern image.

2.4 Camera Pose Estimation

The estimation of the camera exterior orientation for every frame, which defines the camera pose, is accomplished using the pixel coordinates of the corners of the pattern object in the camera frame and their corresponding real-world coordinates as well as the camera interior orientation parameters. These are extracted from the information provided by the camera sensor of the mobile device. The mathematical model used is the projective transformation, as expressed by equation (2).

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} c & 0 & x_0 \\ 0 & c & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2}$$

where x, y = image coordinates of a point in the camera frame corrected by the effects of distortion
X, Y, Z = its real-world coordinates
c = camera constant
x_0, y_0 = principal point coordinates
r_{ij} = the elements of the rotation matrix R
t_i = the elements of the translation vector t
λ = scale factor.

The elements (r_{ij}, t_i) of the joint rotation-translation matrix $[R|t]$ define the camera pose for each frame. They are computed linearly according to the set of equations (3), through computation of the 2D homography H that relates the X, Y real-world coordinates of the pattern object with the corresponding undistorted image coordinates.

$$\begin{aligned} r_1 &= \lambda K^{-1} h_1 \\ r_2 &= \lambda K^{-1} h_2 \\ r_3 &= r_1 \times r_2 \\ t &= \lambda K^{-1} h_3 \end{aligned} \qquad \lambda = \frac{1}{\|K^{-1} h_1\|} \tag{3}$$

where $h_1 = [h_{11} \quad h_{21} \quad h_{31}]^T$
$h_2 = [h_{12} \quad h_{22} \quad h_{32}]^T$
$h_3 = [h_{13} \quad h_{23} \quad h_{33}]^T$
$r_1 = [r_{11} \quad r_{21} \quad r_{31}]^T$
$r_2 = [r_{12} \quad r_{22} \quad r_{32}]^T$
$r_3 = [r_{13} \quad r_{23} \quad r_{33}]^T$
K = the 3×3 camera matrix with the interior orientation parameters
h_{ij} = the elements of the 3×3 homography matrix.

The calculation of the singular value decomposition of the rotation matrix R is accomplished in order to refine it, by coercing it to satisfy the orthogonality condition, as described by Zhang (2000). Then, it is transformed to a 3D rotation vector, using the Rodrigues rotation formula (Kaehler and Bradski, 2017).

Subsequently, the camera pose is optimized via the Levenberg-Marquardt algorithm and the rotation vector is converted back into a 3×3 rotation matrix using the Rodrigues formula. The outcome of this procedure is the joint rotation-translation matrix $[R|t]$ for each camera frame.

2.5 3D Rendering

On real-time pattern recognition, the elements of the estimated joint rotation-translation matrix $[R|t]$ are converted from vectors to a float array and sent to the loading function. They define the current model matrix M_{MODEL} that transforms the vertices of the 3D model from the object coordinate system to their v' position in the world. Model matrix is multiplied by the view matrix M_{VIEW} to convert the world-space coordinates to camera space coordinates. Then, they are multiplied with the projection matrix $M_{PROJ,}$ being normalized in the range [-1, 1] in all three axes. Finally, they undergo a viewport transformation. The screen coordinates v are scaled and translated in order to fit the rendering screen and passed to the rasterization process of the graphics pipeline as a fragment. The transformation of the 3D coordinates in 2D screen coordinates is summarized by the Model View Projection or MVP, as expressed by equation (4).

$$v' = M_{PROJ} \cdot M_{VIEW} \cdot M_{MODEL} \cdot v \quad (4)$$

This sequence of transformations is updated during the drawing calls and applied to the 3D model at each frame. Therefore, the 3D model is visualized on top of the camera stream at the right position, with the intended orientation and scale relative to the pattern object. The rendering of the AR session further involves the construction of the vertex and fragment shaders, the parsing of the 3D file and its material properties as well as the configuration of the lighting and texturing. Specifically, the vertex shader transforms the vertices into screen coordinates through the MVP matrix and the fragment shader calculates the final colour of on-screen pixels. The colour is calculated using the interpolated values passed from the vertex shader to sample from the texture image file. Multiple framebuffers are finally used for the offscreen rendering. In order to prevent any operations overlap between the processing on camera preview and the graphics work, the various tasks are explicitly synchronized.

3. APPLICATION

In this section, implementation details regarding the developed application are presented, the creation of the 3D model of the case study is outlined and the results of the application are discussed. Insights on performance as well as strengths and limitations of the current work are also reported.

3.1 Implementation

The developed AR application is a native Android application built with the OpenCV Computer Vision library and OpenGL ES graphics API. It is based solely on open-source and free-to-use APIs and libraries without any hardware or software dependency. The development environment is Android Studio, integrating both Android Native Development Kit (NDK) for the native C++ implementation and Android Software Development Kit (SDK) for Java programming. The bidirectional communication of the Android SDK and NDK is established by the Java Native Interface (JNI). It handles the calls between C++ functions and Java classes as well as any conversion needed for shared objects. In native side, CMake connects native C++ files with Java and Android SDK. Data structures that represent basic 3D geometry are implemented using the OpenGL Mathematics (GLM) library (GLM, 2021). Besides the construction of the transformation matrices, GLM is used for describing the Wavefront .OBJ format and generating its facet and vertex normal.

The application requires runtime permission to camera and external storage of the device. Once camera access is allowed, home page is loaded and the user can take an image of the point of interest of the real scene which will be augmented (Figure 1). For best performance, the internal environment or object must have enough details and distinguishable characteristics, so that sufficient feature points can be detected. The application stores the captured image as a temporary file in the cache and displays its thumbnail on home screen. The path of the image is sent to the native side for the definition of the pattern object. Then, the application prompts the user to enable the AR functionality. The camera module of OpenCV gets the current frame at run time and sends it through the JNI to the descriptor for features extraction. Every time the camera encounters the captured object, the pattern is detected, image matching occurs and the camera pose is estimated. The resulting rotation-translation matrix is sent back to Java through JNI as a float array. The transformation is needed to set the orientation and finally render the 3D model with OpenGL ES in Java. The captured image acts like a real-world anchor point and the 3D model is visualized at the given pose with fixed position and scale. The system architecture, its basic components along with the functional relationships that accomplish the NDK – SDK connection are presented in Figure 2.

Figure 1. User interface of the application. Left: Home Screen; Right: Camera user interface for pattern image capture.

3.2 3D Model

The object of interest is a 17th century wooden engraved ecclesiastical sanctuary (altar) Ciborium, located in the sanctuary of St. Charalabos church of St. Stephen Monastery, in the archaeological site of Meteora, Greece. It is a form of a wooden engraved canopy supported by columns. It is a relic and distinct structure in the architecture of the Orthodox church, with dimensions: 1.4 m × 1.4 m × 3.0 m. The 3D documentation of this unique artefact along with its AR visualization leads to its dissemination to the public.

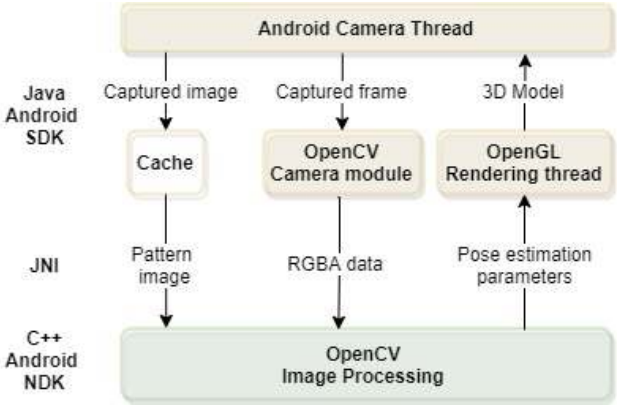

Figure 2. System architecture, technological components and operation process of the AR application.

A total number of 897 images was acquired with Canon EOS 6D camera featuring a 24 mm focal length lens. Structure from motion and dense image matching techniques were applied for the generation of a dense point cloud of the Ciborium using the Agisoft Metashape software (Agisoft, 2021). 3D point cloud editing and meshing were applied using the Geomagic Wrap software (3D Systems, 2021). Finally, texture mapping in the 3D mesh was applied using Agisoft Metashape. The final 3D model, in OBJ format, comprises 30K triangles, corresponding to a size of 4 GB. For the purposes of the mobile application, the number of triangles has been minimized, in order to reduce its size to less than 50 MB. The mesh was decimated without losing the original topology using the pre-built tools of Geomagic Wrap. Details on specific regions of the model were preserved while the whole surface was cleaned and smoothed (Figure 3).

Figure 3. 3D view (left) and zoom-in views (right) of the Ciborium of the church of St. Charalabos.

3.3 Results

As far as the pattern recognition results are concerned (Figures 4, 5), the matching ORB feature points detected by the application lead to successful pattern object recognition, in the case that it is depicted in approximately the same scale with its scale in the pattern image, regardless of any rotation with respect to the pattern image. As also stated in the original paper of the ORB detector (Rublee et al., 2011), whereas the extracted feature points are rotation invariant, the issue of scale invariance has not been adequately addressed. In spite of the fact that ORB uses a pyramid scheme, issues like scale per keypoints or implementation of octaves are not tackled by the detector. Hence, it is not robust enough in changes of scale. This issue may be overtaken by the application by taking a pattern image that corresponds to approximately the same scale with the scale of the pattern object in each frame. However, taking into account the fact that scale invariance is a necessary characteristic in AR applications, future work will investigate other feature detectors, robust in changes of scale, like SURF or SIFT.

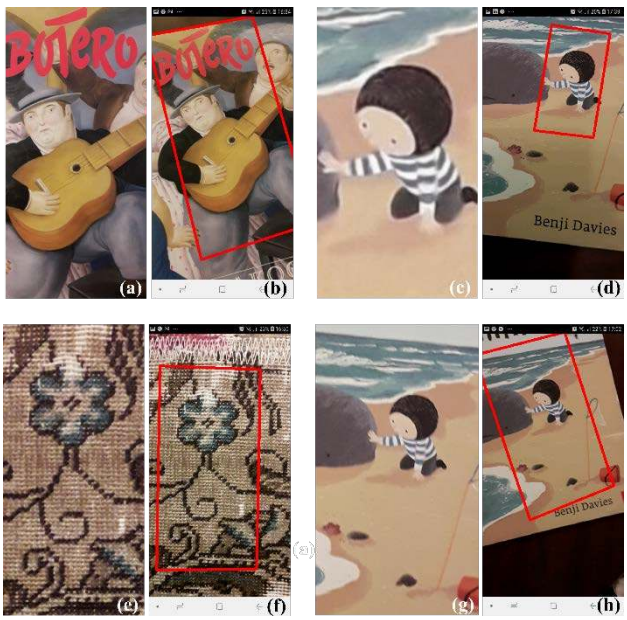

Figure 4. Recognition of four different pattern objects; (a), (c), (e), (g): pattern images captured by the user; (b), (d), (f), (h): pattern recognition performed by the application.

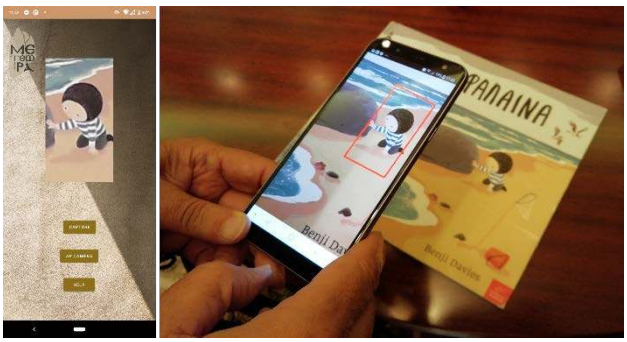

Figure 5. Left: User interface, displaying the pattern image captured by the user; Right: Photograph showing the pattern recognition process by the application.

Regarding the AR session, an accurate and qualitative overlay of the 3D model of Ciborium at the given pose is achieved (Figures 6, 7). The level of the display resolution is good, regarding the complexity of the surface. It can be observed that the different configurations on diffuse and specular maps that determine how the material is rendered in light, simulate different lighting conditions. The 3D model is attached at the center of the physical object, but tracking loses its efficiency during fast camera movements.

The performance evaluation is conducted using Android profiler tools. The AR activity is being recorded three times using the same pattern image and Android smartphone device for benchmarking validation. The device displays 60 frames per seconds (fps) and optimal performance is attained when every frame is rendered in less than 16.67 ms. If rendering time exceeds this limit, a frame drop occurs. Table 1 reports frame counts and

latency at three critical moments: on pattern recognition, on 3D model loading and on final display of 3D model.

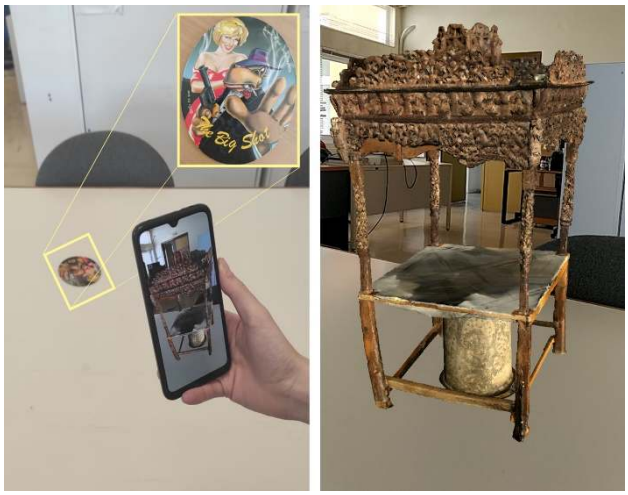

Figure 6. Different views during the AR session of the application. Left: Photograph showing the pattern image and the screen display of the 3D overlay; Right: The 3D model of Ciborium as it is superimposed on top of the object.

Figure 7. The 3D model of the Ciborium overlaid on top of a box.

It can be observed that the application manages to deliver an average of 52 fps but drops frames during loading as well as at drawn time. The time spent between pattern detection and recognition is insignificant. However, the user waits 5.54 sec before seeing the 3D overlay. To investigate which rendering phases or events cause the latency and the framerates drops, Android trace analysis tool is exploited. Tracing data are collected for both Java and Native code execution during this time period. Two metrics are explored: GPU activity and CPU usage. The *onDrawFrame* function reports a long delay on loading, which is caused by the size of the mesh and its texture. A high number of vertices is submitted by the application and long time is recorded during waiting for dependent memory accesses, such as texture sampling. The whole evaluation indicates that the application succeeds in delivering a good frame latency distribution and maintaining the GPU at a consistent performance. Regarding computing workload, marker-less AR is a CPU-intensive task with increased memory consumption (Figure 8). The fact that real-time processing occurs on native side compensates for any delay on camera thread, providing a significant reduction in CPU overhead.

Table 1. Metrics on UI thread, namely the main thread, at pattern recognition, loading and drawn time.

	Pattern recognition	Loading	Display
UI (FPS)	60	42	53
Delay (sec)	0.3	5.54	

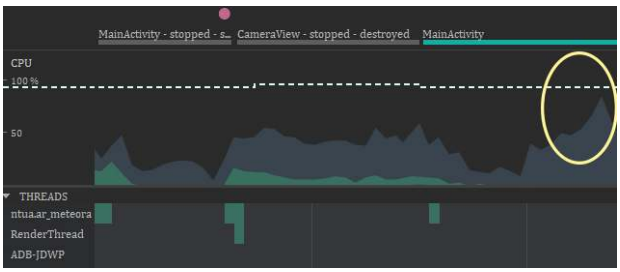

Figure 8. Graph of real-time CPU usage while the user captures the pattern image (CameraView activity) and enables the AR functionality (MainActivity activity). The increased CPU usage at the moment of pattern recognition is highlighted.

4. CONCLUSIONS AND FUTURE WORK

This paper presents the development of a content-based AR application for Android mobile platforms, which overlays high-resolution 3D models at predefined locations of the real environment of the user. The application has been developed using pure computer vision and photogrammetric techniques, in C++ and Java programming languages. It is built with the use of the OpenCV library and the OpenGL graphics API without any dependencies on AR SDKs. Compared with other recent MAR applications in Cultural Heritage, our approach does not require any specialized software to be downloaded by the user in order to operate, having no hardware dependencies. It aims at evolving the way archaeological, historical and cultural assets are accessed and displayed. The application is flexible, on the grounds that it may be applied in numerous case studies and 3D augmentation objects, ranging from immovable cultural relics and artefacts to CH monuments and even 3D models that may not have any cultural value. This flexibility is due to the pattern-based AR methodology followed in the application, which relies on the recognition of the proper pattern in the real-world scene. The planar pattern object along with the 3D augmentation model are defined as input data in the application, which thus offers unlimited functionality regarding the case studies.

The application is still under development, for being optimized in terms of reliability of pattern object recognition and speed, which are the most fundamental issues for computationally

intensive real-time marker-less AR augmentation. Currently, the AR tracking is not yet well-optimized. The application corresponds to successful pattern object recognition in the case that the pattern is depicted in approximately the same scale with its scale in the pattern image, regardless of any rotation with respect to it, due to the used ORB detector. Future work will investigate the performance of other feature point detectors, robust in changes of both scale and orientation, like SURF and SIFT, so that the tracking of the pattern object becomes more reliable, independently from its scale and orientation in the camera frames. The best solution in terms of accurate localization of the pattern object and fast image matching will be found, in order to combine both reliability and effectiveness. Moreover, the application calls the whole OBJ model and texture files at run-time, instead of implementing progressive loading or a LoD (Level of Detail) representation of geometry. Future work will investigate such techniques, in order to reduce memory consumption by the computationally intensive task of run-time loading of the high-resolution 3D textured model.

ACKNOWLEDGEMENTS

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competiveness, Entrepreneurship and Innovation, under the call RESEARCH–CREATE–INNOVATE (project code: T1EDK-02859).

REFERENCES

Agisoft, 2021. Agisoft Metashape Software. https://www.agisoft.com/ (20 April 2021).

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008. Speeded-up robust features (SURF). *Computer Vision and Image understanding*, 110(3), 346-359.

Caarls, J., Jonker, P., Kolstee, Y., Rotteveel, J., van Eck, W., 2009. Augmented reality for art, design and cultural heritage: system design and evaluation. *EURASIP J. Image Video Process.*, 1-17.

Calonder, M., Lepetit, V., Strecha, C., Fua, P., 2010. BRIEF: Binary robust independent elementary features. In *Computer Vision – ECCV 2010*. LNCS, vol 6314. Springer, Berlin, Heidelberg. doi.org/10.1007/978-3-642-15561-1_56.

Čejka, J., Zsíros, A. Liarokapis, F., 2020. A hybrid augmented reality guide for underwater cultural heritage sites. *Personal and Ubiquitous Computing*, 24, 815-828. doi.org/10.1007/s00779-019-01354-6.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6), 381-395.

Galatis, P., Gavalas, D., Kasapakis, V., Pantziou, G.E., Zaroliagis, C.D., 2016. Mobile Augmented Reality Guides in Cultural Heritage. *MobiCASE*, 11-19.

OpenGL Mathematics (GLM), 2021. https://github.com/g-truc/glm (20 April 2021).

Haugstvedt, A.C., Krogstie, J., 2012. Mobile augmented reality for cultural heritage: A technology acceptance study. *ISMAR 2012*, 247-255.

Kaehler, A., Bradski, G., 2017. *Learning OpenCV 3: Computer vision in C++ with the OpenCV library*. O'Reilly Media, Inc.

Khan, M.A., Israr, Almogren, A.S., Din, I.U., Almogren, A., Rodrigues, J.J.P.C, 2021. Using augmented reality and deep learning to enhance Taxila Museum experience. *J Real-Time Image Proc*, 18, 321-332. doi.org/10.1007/s11554-020-01038-y.

Khronos Group, 2021. OpenGL. https://www.opengl.org// (20 April 2021).

Kunjir, A.R., Patil, K.R., 2020. Effectiveness of practicing social distancing in museums and art galleries for visitors using Mobile Augmented Reality (MAR): SMART - Social distancing using Mobile Augmented Reality technology. *Int. J. Art, Cult. Design Technol.*, 9(1), 1-14. doi.org/10.4018/IJACDT.2020010101.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Computer Vision*, 60(2), 91-110.

Moré, J.J., 1978. The Levenberg-Marquardt algorithm: implementation and theory. *Numerical analysis*, 105-116. Springer, Berlin, Heidelberg.

OpenCV Team, 2021. OpenCV. https://opencv.org/ (20 April 2021).

Panou, C., Ragia, L., Dimelli, D., Mania, K., 2018. An architecture for mobile outdoors augmented reality for cultural heritage. *ISPRS Int. J. Geo-Inf.* 7(12), 463. doi.org/10.3390/ijgi7120463.

Papagiannakis, G., Ponder, M., Molet, T., Kshirsagar, S., Cordier, F., Magnenat-Thalmann, N., Thalmann, D., 2002. LIFEPLUS: Revival of life in ancient Pompeii. *VSMM 2002*.

Ramtohul, A., Khedo, K.K., 2019. A Prototype Mobile Augmented Reality Systems for Cultural Heritage Sites. In *Information Systems Design and Intelligent Applications. Advances in Intelligent Systems and Computing*, vol 863. Springer, Singapore. doi.org/10.1007/978-981-13-3338-5_17.

Rosten, E., Drummond, T., 2006. Machine learning for highspeed corner detection. In *Computer Vision – ECCV 2006*. LNCS, 3951. Springer, Berlin, Heidelberg. doi.org/10.1007/11744023_34.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to SIFT or SURF. *ICCV 2011*, 2564-2571. doi.org/10.1109/ICCV.2011.6126544.

Siriwardhana, Y., Porambage, P., Liyanage, M., & Ylinattila, M., 2021. A Survey on Mobile Augmented Reality with 5G Mobile Edge Computing: Architectures, Applications and Technical Aspects. *IEEE Communications Surveys & Tutorials*. doi.org/10.1109/COMST.2021.3061981.

Trunfio, M., Lucia, M.D., Campana, S., Magnelli, A., 2020. Innovating the cultural heritage museum service model through virtual reality and augmented reality: the effects on the overall visitor experience and satisfaction. *Journal of Heritage Tourism*. doi.org/10.1080/1743873X.2020.1850742.

Vanoni, D., Seracini, M., Kuester, F., 2012. ARtifact: Tablet-Based Augmented Reality for Interactive Analysis of Cultural Artifacts. *2012 IEEE International Symposium on Multimedia*, 44-49. doi.org/10.1109/ISM.2012.17.

Venigalla, A.S.M., Chimalakonda, S., 2019. Towards enhancing user experience through a web-based augmented reality museum. *2019 IEEE 19th Int. Conf. on Advanced Learning Technologies (ICALT)*, 357-358. doi.org/10.1109/ICALT.2019.00110.

Verykokou, S., Ioannidis, C., Kontogianni, G., 2014. 3D visualization via augmented reality: The case of the Middle Stoa in the Ancient Agora of Athens. In *Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection. EuroMed 2014*. LNCS, vol 8740, 279-289. Springer, Cham. doi.org/10.1007/978-3-319-13695-0_27.

Vlahakis, V., Ioannidis, N., Karigiannis, J., Tsotros, M., Gounaris, M., Stricker, D., Gleue, T., Daehne, P., Almeida, L., 2002. Archeoguide: An Augmented Reality Guide for Archaeological Sites. *IEEE Comput. Graph. Appl.*, 22, 52-60.

Zhang, Z., 2000. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11), 1330-1334.

Zoellner, M., Stricker, D., Bleser, G., & Pastarmov, Y., 2007. iTACITUS – novel interaction and tracking paradigms for mobile AR. *VAST 2007,* 26-30.

Zhu, Y., & Li, N., 2021. Virtual and augmented reality technologies for emergency management in the built environments: A state-of-the-art review. *J. Safety Science and Resilience*, 2(1), 1–10, doi.org/10.1016/j.jnlssr.2020.11.004.

3D Systems, 2021. Geomagic Wrap Software. https://www.3dsystems.com/software/geomagic-wrap (20 April 2021).