DOI https://doi.org/10.24297/ijct.v22i.9335

A Technique to Estimate the Mobile Position and Rotation

Trinh Hien Anh [1], ***Trinh Xuan Hung*** [1],***Ha Manh Toan*** [1]

[1] **Researcher, Institute of Information Technology, VAST**

Email: hienanh@ioit.ac.vn, trxhung@ioit.ac.vn, hmtoan@ioit.ac.vn

Abstract

Nowadays, almost all applications, especially augmented reality (AR) applications, run on the web. In these applications, determining the position and rotation of the mobile device is an indispensable basic step. The speed and accuracy of this work greatly affect the quality of the user experience. Therefore, estimating the rotational position of mobile devices on the Web is necessary and meaningful in practice. In this paper, we propose a way to estimate the position and rotation of the mobile device using the image of the marker obtained from the camera combined with the data from the device's tilt angle sensor. The proposal has been experimentally installed and evaluated for performance on Web Assembly, Java Script, and C++ platforms. Along with that, to ensure objectivity, we compared the speed and calculation error of the proposed technique with the P3P and PnP techniques installed in the OpenCV open-source library. We also use the proposal method to develop the Virtual Museum application for the Vietnam National Museum of Nature

Keywords: AR, Camera pose, Marker detect, VR.

Introduction

Multimedia applications in general, and virtual reality and augmented reality in particular, have been playing an important role in the development of information technology in recent times. They provide new, unique, and intuitive ways to present data to users. In particular, the versatility of mobile devices, typically mobile phones, has provided many possibilities for the development of applications using these technologies. For example, in the Pokemon Go game application, the synchronization between the virtual world and the geospatial space of the real world has brought impressive experiences to players. The success of this application has created a new development trend for many other practical applications. Or a more common example is Google Maps (Fig.1). When using the navigation function, the application will synchronize the user's location with the map data in the database to give appropriate directions along with realistic visual images to the user.

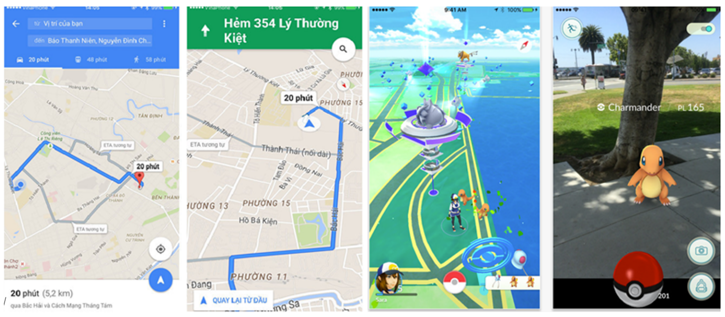


Figure 1: Google map and Pokemon Go

In many applications, the problem of estimating the position and rotation of mobile devices plays a fundamental role in creating impressive user experiences. For example, with some applications for virtual tours of galleries, where the user can move around the room and observe the whole gallery with the item in it. With each item, the user can view all the attached information visually right at the observation location. Moreover, users can observe items that are not in the gallery, which can even move, thereby creating an impressive experience for visitors. In such cases, calculating the position and rotation of the receiver, possibly a mobile phone, plays a particularly important role in providing input to the image rendering process. can produce results that give spatial sensations between the user and the virtual world.

Researcher are pay attention on estimating position and rotation angle of the mobile device. (*Camposeco, F., Cohen, A., Pollefeys, M., &Sattler*) proposed a hybrid model instead of simply using the "2D-3D" or traditional "2D-2D" in case of structure from motion. They improved RANSAC method (Fig.2).

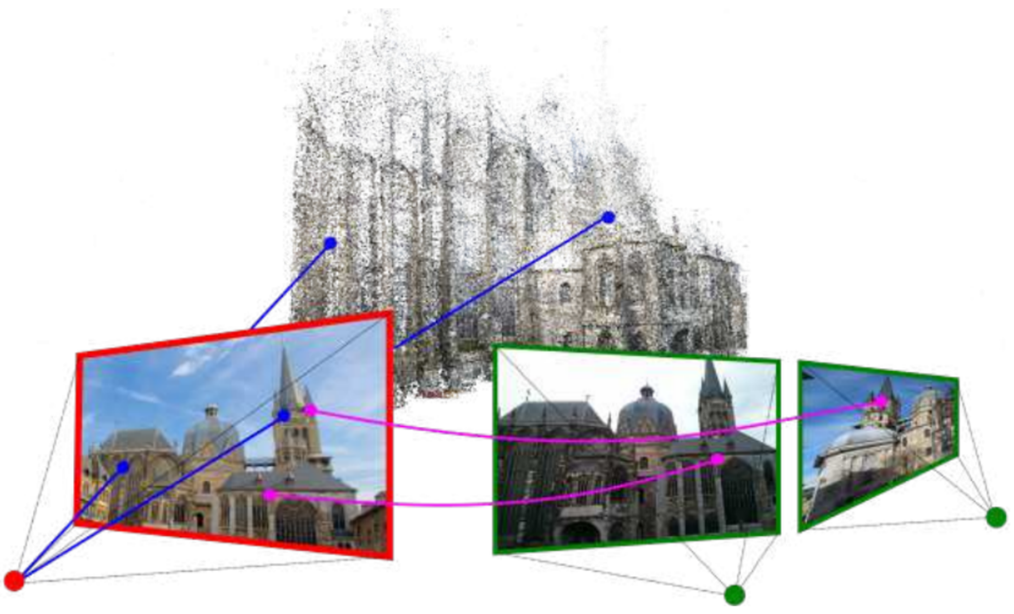


Figure 2: Camposeco's proposal: pink: 2D-2D matching, blue 2D-3D matching

The input that can be processed by 2D-3D or 2D-2D matching techniques as proposed by the Camposeco team is a set of 2D landmarks on the image. With structure from motion systems, we often need invariant feature points on the image, such as SURF (*Bay, H., Tuytelaars, T., &Van Gool, L.*), SIFT (*Lowe, D. G.*),... However, in a stable environment or for those needing additional information for the observation environment, using markers is the best choice. In these environments, designing a marker attracts a lot of research attention. For example, Romero et al. (*Romero-Ramire, F. J., Muñoz-Salinas, R., &Medina-Carnicer, R..*) introduced a fractal-based marker. This helps to deal with many limited situations, such as partially obscured markers or where the processing quality depends too much on the size. In addition to the design pattern, they also presented a corresponding positioning technique to be able to work around these limitations (Fig.3).

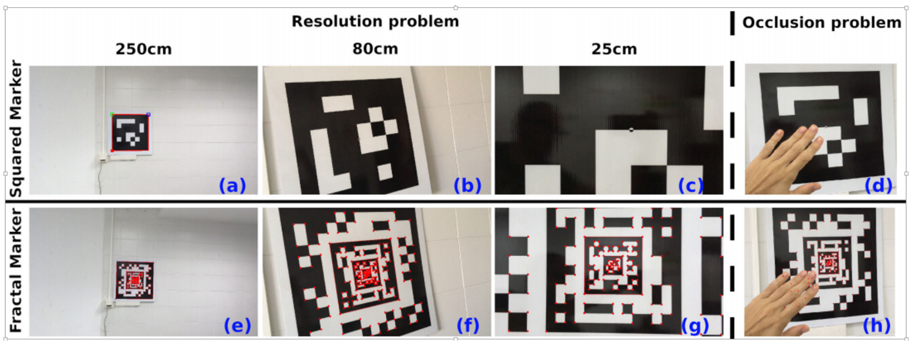


Figure 3: Romero's example

The W3C is developing a Web XR specification (*W3C Candidate Recommendation Draft, 24 August 2022*).This specification supports virtual reality and augmented reality features on websites. This is a strong testament to the growth of this field. However, at the moment, Web XR is still an experimental draft with many limitations in terms of the features that can be accessed. With web developer... This made pure JavaScript-based computing systems difficult at times, necessitating accelerated solutions such as GPUs and Web Assembly.

This paper presents a technique to estimate the position and rotation of the mobile. The technique is based on calculating the relation between points in 3D space and the points in the marker. Further, we use the real data from the rotation sensor to calculate the equipment's rotation. We did the experiment on the HTML 5 platform. With the aim of enhancing performance, we use Web Assembly technology.

Estimate the position and rotation of the mobile

1. Design marker

We use anchor points to define the correlation between the real and virtual worlds. These points contain the spatial information and other constraints. From this points, the system identify object in the real world. We can calculate anchor points with the information from the sensor, camera sensor. Usually, the corner points of the marker are anchors.

In this proposal, the rotation of the mobile device can be calculated based on the data of the point set mapping (2D–3D) or used directly from the real-time data provided by the mobile device's rotation sensor. The marker is specifically designed to make the construction of the processing algorithm simpler and more efficient. In this case, the samples are black and white interlaced tables. The black outermost border contains no information. Four squares in the four corners, including 3 black squares and 1 white square, are responsible for orienting the marker. The remaining cells are information cells and contain the same number of checksums, each corresponding to one bit. The checksum spread is customizable, usually 8 or 16 squares. The internal data also undergoes a simple encryption step by performing XOR with a predefined key to increase the identity security of the marker object, as shown in the example in (Fig.4).

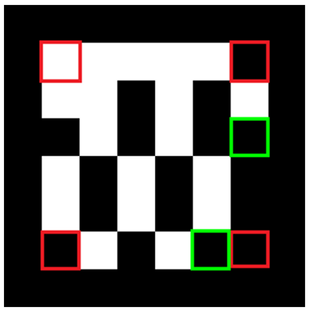


Figure 4: 6x6 marker with the encode information: 4-oriented (red) and 2-checksum (green)

2. Determine the anchor point

The anchor points in this case are understood as the vertex set of the marker object in the image and the corresponding marker object vertices in the three-dimensional space of the real world. Determining this set is equivalent to the problem of locating and identifying objects marked in the image. Figure 5 below shows the main steps of the algorithm to locate and identify the marker object in the image.

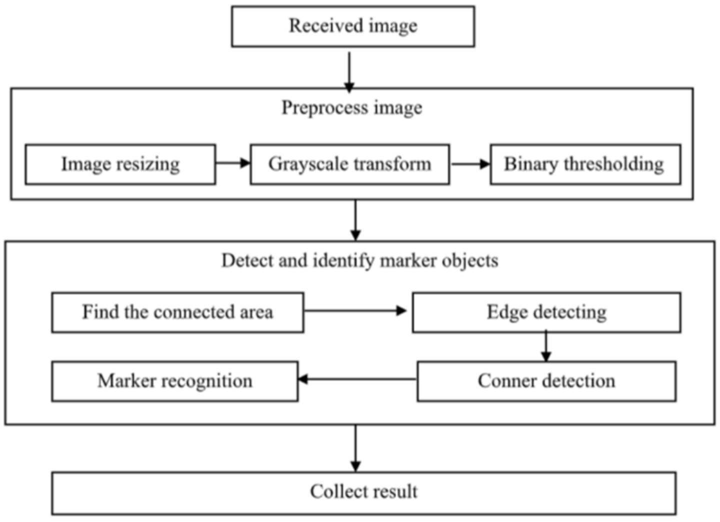


Figure 5: Detect and recognize marker objects

After the image is obtained from the camera, we perform two main phases: the first is image pre-processing; the second is detecting and identifying the marker object to obtain the position of the vertices of the marker on the image and the identification of the marker. Finally, we combine the results to return the set of vertex coordinates of the marked object on the image and the corresponding coordinate values in real world three-dimensional space. In which the vertex values of the marker in three-dimensional space are stored in advance in the database according to the identifier of the marker.

The marker is black and white with a simple texture. The design aims to make segmentation easier. The semantic analysis of the image area to consider the marked object will frequently be based on the criteria of the region's local texture rather than the object's colour features. Similarly, the acquired images will be converted to gray-scale for processing. This is also a popular image format used by many image processors. Under certain environmental conditions, such as lighting, background colour, and so on, selecting an effective image graying formula can help to some extent in enhancing marker information and reducing the external influence of the environment.

The next step is to perform threshold on the gray-scale image. Basically, assuming that the marker image consists of 2 regions of black and white images, we expect that the threshold result will also divide these two regions well. In this case, after division, pixels with high intensity values, including white areas of the marker object, will be labelled as white areas on the resulting binary image; and pixels with low intensity values, including the black areas of the marker object, which will be labelled as black areas on the resulting binary image. In our research, we have deployed and tested a few threshold algorithms similar to those proposed in the publications [9->12]. Preliminary experiments have shown that the threshold algorithm in Sauvola's publication (*Sauvola, J., & Pietikäinen, M.*) has a high agreement with the data as in our work.

Binary images will be processed in such a way that the relevant regions are spread according to a set of criteria: 4-neighbor or 8-neighbor. At that point, some closed and distinct areas will appear in the binary image. The splashing process will go through each pixel of the binary image, looking for a communication relationship to label the area. This procedure is carried out sequentially. We'll finish some binary image filling areas. Interconnection zones will also be filtered and processed to eliminate some areas that are too small, frequently noisy, and unrelated to the analysis, which can quickly eliminate some areas that are too large to be the area of interest.[5pt] Then, what needs to be dealt with is extracting the borders that surround the selected inter-connectors and calculating the angle points of the marked object. In this step, we look for lines parallel to the Hough transform, then we analyse the length and find the intersection to take the angle points.

From the calculated corner points, we can estimate the Homography transformation with the aim of converting the quadrilateral created by the 4 corner points into the square corresponding to the marker. Applying this transformation to the polygon, we will get a square image.

The next step is to decode the received image to get the identification of the marker. The decoding is done by looking at each image area according to the original design of the marker, calculating the ratio between the black pixel and the white pixel. Based on this ratio, we decide that the square has bit 0 or bit 1; combining the data bits together to get a bit sequence; next, checking to ensure the correctness of the data series based on the checksum. The final step is to combine and decode the data bits to obtain the identifier of the marker.

3. Refine the anchor

The landmarks on the estimated mark on the image often have certain errors due to environmental influences and variations during processing. This error significantly affects the accuracy when we estimate the position and angle of the camera. The purpose of this step is to micro-adjust the angle points of the marked object to achieve the most accurate position. In the previous step, we learned the location of the tops of the marked objects on the image; we also know that the identity of the mark is synonymous with knowing the vector image of the marker. On the other hand, we can represent the error function of the specified mark-up object on the image as follows:

$$\int\int_{(x,y)\in A} I_{x,y} - {I^0}_H(x,y)$$

In which: A is the area formed by the four peaks of the object marked on the photograph; (x, y)is the number of entities representing the point coordinates in image space; and $I_(x,y)$is the pixel value at point (x, y) calculated by interpolation of four neighbouring points on the raster image; $I_((x,y))^0$ pixel value at coordinate x, y on vector $imageI^0$; H is the Homography matrix to transform the four points of the marker in the camera space.Estimating H is quite simple when we know the coordinates of the four peaks of the mark on the image.

Our job is to conduct the top correction of the mark object on the image so that the error function value is the smallest. This can be done by turning microscopic the tops of the marker on the image to find the values so that E reaches the maximum; or you can also use gradient repeating techniques to find the top values of the marker so that E reaches its maximum. In the installation, they use the Gauss-Newton repeating technique (*Gratton, Serge, Amos S. Lawless, and Nancy K. Nichols.*) to find these peaks.

4. Position and rotation estimation

In the originally set virtual world, each landmark on the camera image identified in the previous step will have a corresponding 3D location (based on the identity of the marker). What we need to do is calculate the position and rotation of the camera in the virtual world so that the image obtained from it matches the image of the real camera. More specifically, the landmark locations will match.

Mathematically, at each calculation step, we have a 3D score set $\{P_i\}_{i=1}^{n}$ in the virtual world and a 2D score set $\{Q_i\}_{i=1}^{n}$ camera photos. We need to estimate the displacement parameters t_x, t_y, t_z and dial parameters $\theta_x, \theta_y, \theta_z$ to set 3D points$\{P_i\}_{i=1}^{n}$ after performing the projection corresponding to those two sets of parameters will result in a 2D score set $\{Q_i\}_{i=1}^{n}$ We build translation matrix:

$$T = \begin{vmatrix} t_x \\ t_y \\ t_z \end{vmatrix}$$

and the rotation matrix:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta_x & sin\theta_x \\ 0 & -sin\theta_x & cos\theta_x \end{bmatrix} * \begin{bmatrix} cos\theta_y & 0 & -sin\theta_y \\ 0 & 1 & 0 \\ sin\theta_y & 0 & cos\theta_y \end{bmatrix} * \begin{bmatrix} cos\theta_z & -sin\theta_z & 0 \\ -sin\theta_z & cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Transform the 3D set $\{P_i\}_1^n$ by T and R:

$$P`_i = R * P_i + T.$$

The result of the projection:

$$Q`_i = \left( \frac{P`_{i_x} * focal}{P`_{i_z}}, \frac{P`_{i_y} * focal}{P`_{i_z}} \right)$$

In which, focal is the camera focal, it is assumed. The setof 2D $\{Q_i\}_1^n$ need to meet with the set $\left\{Q`_i\right\}_1^n$, we do this through minimization the error function $E = \sum_{i=1}^{n} distance(Q_i, Q`_i)$. In the paper, the calculation is done by steepest descent method(*Debye, P.,*).

5. Update with the mobile parameters

The proposed technique is aimed at mobile devices, which are usually equipped with rotation sensors that provide real-time tilt of the device relative to the earth. We therefore provide a function, with which we can take advantage of the rotational angle data from the device as input for the calculation process.

For consistency across different platforms, we follow the reference from the W3C Orientation Sensor (*W3C Candidate Recommendation Draft, 24 August 2022*) specification. The specification represents the basic interfaces of the rotation sensor, through which one can query in real time information about the physical rotation of the device with the stationary Cartesian coordinate system of the earth. The output data format is also described as being compatible with the WebGL platform.

Basically, we can visualize the computational context as consisting of the following components: the real world, the virtual world, and the camera. In which, the real world is fixed – this is the understood world whose coordinate system corresponds to the stationary Cartesian coordinate system of the earth described in the Orientation Sensor specification. This coordinate system is fixed and does not change during program execution. The virtual world is the space we put 3D objects in, it contains 3D points as explained in the above section. The virtual world is preset because we need to arrange it so that it corresponds to a space in the real world. Thus, this coordinate system is also fixed and does not change during program execution. Therefore, the transformation of an object from the virtual world to the real world or vice versa is fixed. The matrix of this transformation can be computed once and reused in the process. The third component is the camera or we can uniformly call it the mobile device. From the parameters in the specification (*Y.K. Lai, P.L. Rosin*), we can get the camera's tilt angle relative to the real world. Thus, by bridging, we can calculate the camera's tilt angle in the virtual world, which corresponds to the parameter set $\theta_x, \theta_y, \theta_z$ described in the previous step. Thus, by updating with the parameters of the mobile device, we have reduced the set of rotational parameters that need to be calculated thereby increasing the speed and accuracy of the algorithm.

Experiment

1. Evaluate the performance of the position and rotation angle estimating algorithm

In this test, we install the algorithm to estimate the position and camera angle when knowing the position of the 4 vertices of the marker. With the assumption of position, the camera's rotational angle varies continuously with time, which means that the deviations of these prices are small at two adjacent sampling times. The algorithm is installed on 3 platforms: 1 is Native C++, 2 is JavaScript, and finally Web Assembly. Each platform is implemented with two options: 1) don't know the camera angle value; and 2) know the camera angle value in advance. Along with that, we also built test programs using P3P and PnP algorithms installed in OpenCV on a native C++ platform.

The input data set is generated by projecting the 4 vertices of the marker onto the camera observation plane when the position and rotation angle of the camera change randomly and continuously over time. Thus, at each sampling time, we always know the position and rotation of the camera in 3D space, and we also know the position of the vertices of the marker on the observation plane. By such sampling, we create 1,000,000 sample data sets. Each dataset consists of two components: one is the camera's position and rotation angle; two is the position of the 4 vertices of the sampler in the camera projection plane.

The evaluation of algorithms is considered on two criteria: one is the execution time; the other is the average position error and the mean rotation angle error. The following table summarizes the results of the performance and error evaluation of the algorithms, in which: C++: is the algorithm installed in C++ and running on the desktop; JavaScript is an algorithm that runs in a Web environment and is implemented by JavaScript. Web Assembly is a C++ program that compiles Web Assembly code and runs it in a Web environment. Similarly, the case of knowing the rotation angle in advance and only needing to estimate the object position will be denoted by R-C++, R-JavaScript, and R-Web Assembly.

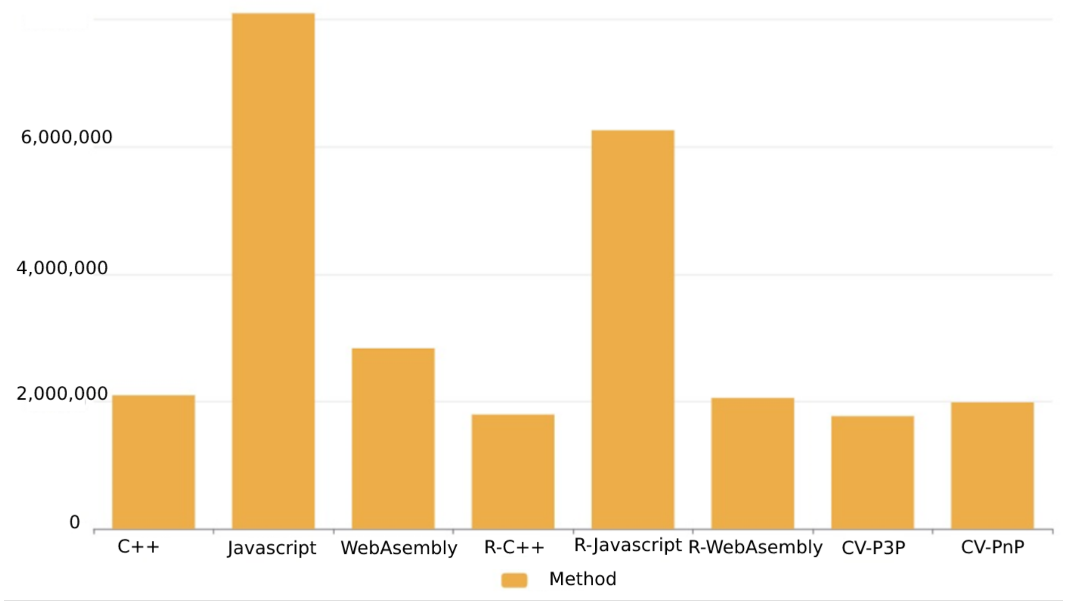


Figure 6: Execution time result

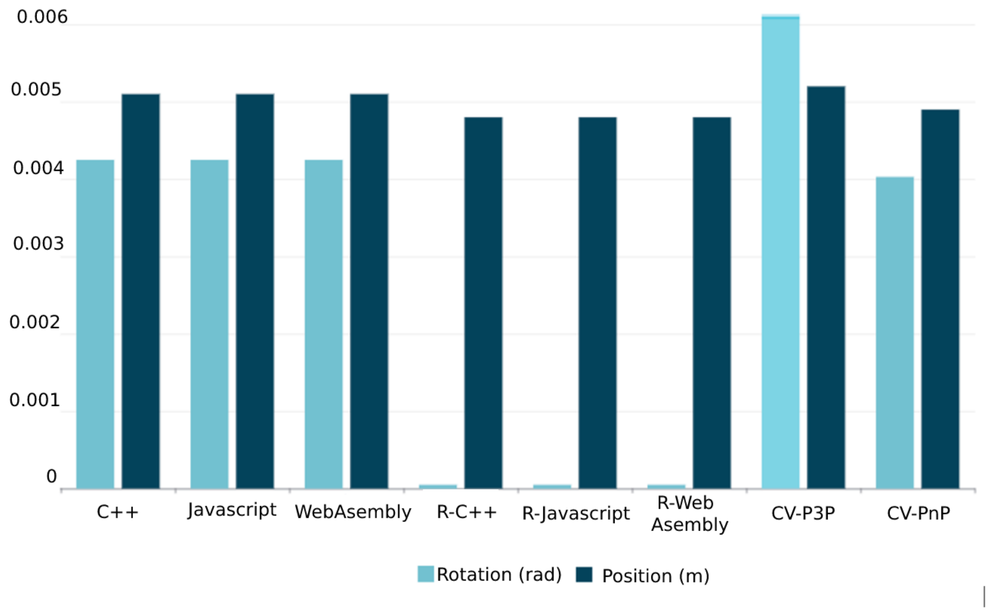
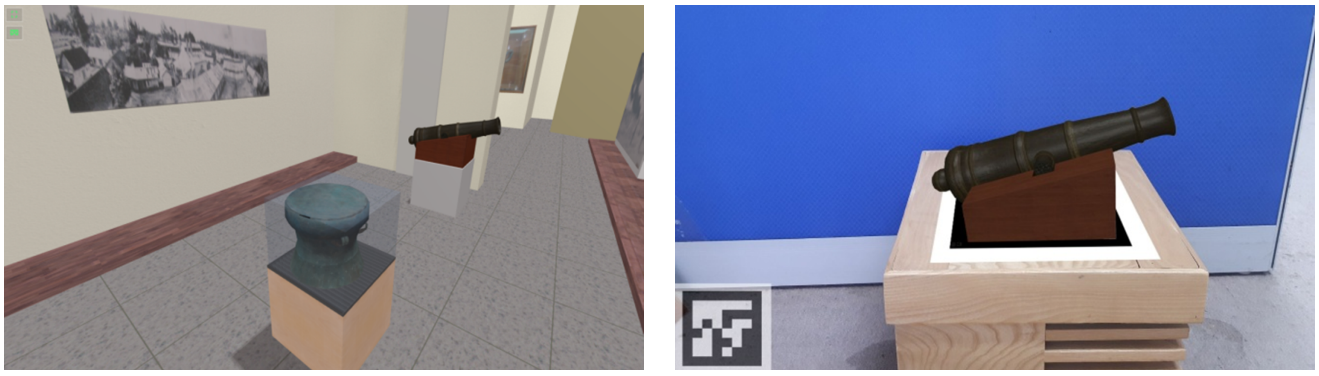


Figure 7: Wrong summary table calculates position, angle of rotation

Based on analysis of two graphs (Fig. 6, 7) of execution time and error during implementation, we have the following conclusions: One is that the algorithm (not knowing the rotation angle in advance) is effective in terms of execution time and average error. OpenCV; the average of the PnP algorithms installed in OpenCV; Second, in the case of knowing the rotation angle in advance, the algorithm proves to be more efficient in all aspects than the techniques implemented in P3P and PnP; Algorithms when implemented using Web Assembly give much better execution time than those implemented in JavaScript and are roughly equivalent to algorithms implemented in Native C++. Thus, it can be concluded that the approach of the study is appropriate and effective.

2. Experiment in application

The entire technique presented in the article has been installed by us and tested in the virtual display support system with two visiting scenarios approached in two directions: AR and VR. With enhanced virtual reality, we assign objects to each object marked according to the identity of the mark out. The marker is printed and pasted in the area where the AR experience is needed (Fig. 8-b). Users can experience three dimensional images and information of virtual display artefacts by hand-phone. With the joint VR application (Fig. 8-a), I organize a physical space in which the marked objects are pasted. Each marker is positioned by measuring its position and angle of rotation in space. Then the user can use a virtual reality headset to observe the virtual environment. Location, angle of view, and other parameters will be estimated and mapped from space according to the techniques presented above.



Experiment in virtual museum app Experiment in augmented reality

Figure 8: Experiment in aplications

Conclusion

Virtual reality and augmented reality have been making strong developments that have contributed to changing the appearance of the information technology sector for social life. The paper presents a technique that captures the position and rotation of mobile devices and, more specifically, the camera of the device, which still makes a lot of sense both in terms of theoretical research and implementation practices of virtual reality and augmented reality. The technique has been tested and applied in the virtual museum's display system, which is also a central research direction of the team to bring the research results to real life.

References

Bay, H., Tuytelaars, T., & Van Gool, L. "Surf: Speeded up robust features". In European conference on computer vision. Springer, Berlin, Heidelberg, pp. 404-417, 2006.

Camposeco, F., Cohen, A., Pollefeys, M., Sattler, "T. Hybrid camera pose estimation". in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 136-144, 2018.

Damera-Venkata, Niranjan, and Brian L. Evans. "Adaptive threshold modulation for error diffusion halftoning." IEEE Transactions on Image Processing 10.1 (2001): 104-116.

Debye, P., "Näherungsformeln für die Zylinderfunktionen für große Werte des Arguments und unbeschränkt veränderliche Werte des Index", Mathematische Annalen, 67(4), pp. 535–558, 1909.

Gratton, Serge, Amos S. Lawless, and Nancy K. Nichols. "Approximate Gauss–Newton methods for nonlinear least squares problems". SIAM Journal on Optimization 18.1, pp.106-132, 2007

Lowe, D. G. "Distinctive image features from scale-invariant keypoints". International journal of computer vision, 60(2), pp.91-110, 2004.

M. Luessi, M. Eichmann, G. M. Schuster, and A. K. Katsaggelos, Framework for efficient optimal multilevel image thresholding, Journal of Electronic Imaging, vol. 18, pp. 013004+, 2009. doi:10.1117/1.3073891.

Romero-Ramire, F. J., Muñoz-Salinas, R., & Medina-Carnicer, R.. "Fractal Markers: a new approach for long-range marker pose estimation under occlusion". IEEE Access, 7, pp.169908-169919, 2019.

Sauvola, J., & Pietikäinen, M. (2000) Adaptive document image binarization. Pattern Recognition, 33(2), 225–236. doi:10.1016/s0031-3203(99)00055-2

Y.K. Lai, P.L. Rosin, Efficient Circular Thresholding, IEEE Trans. on Image Processing 23(3), pp. 992–1001 2014). doi:10.1109/TIP.2013.2297014.

W3C Candidate Recommendation Draft, 24 August 2022 Accessing virtual reality (VR) and augmented reality (AR) devices, including sensors and head-mounted displays, on the Web. [Online] . Available: https://www.w3.org/TR/webxr.

W3C Working Draft, 2 September 2021 Base orientation sensor interface and concrete sensor subclasses to monitor the device's physical orientation in relation to a stationary three dimensional Cartesian coordinate system. [Online]. Available https://www.w3.org/TR/orientation-sensor.

Conflicts of Interest

Authors declare all that there is no conflict of interests.

Funding Statement

Funding Statement: This research was funded by the DTCS-2108.