J. Vis. Commun. Image R. 37 (2016) 63–69

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

An improved augmented reality system based on AndAR ☆

Peng Chen *, Zhang Peng, Dalong Li, Lijuan Yang

College of Computer and Information Technology, China Three Gorges University, Yichang, China

ARTICLE INFO

Article history:
Received 1 January 2015
Accepted 29 June 2015
Available online 4 July 2015

Keywords:
AndAR
Mobile augmented reality
Android platform
Registration
Inliers tracking
Feature extraction and matching
Low-level visual features
Architecture

ABSTRACT

AndAR is a project applied to develop Mobile Augmented Reality (MAR) applications on the android platform. The existing registration technologies of AndAR are still base on markers assume that all frames from all videos contain the target objects. With the need of practical application, the registration based on natural features is more popular, but the major limitation of the registration is that many of them are based on low-level visual features. This paper improves AndAR by introducing the planar natural features. The key of registration based on planar natural features is to get the homography matrix which can be calculated with more than 4 pairs of matching feature points, so a 3D registration method based on ORB and optical flow is proposed in this paper. ORB is used for feature point matching and RANSAC is used to choose good matches, called inliers, from all the matches. When the ratio of inliers is more than 50% in some video frame, inliers tracking based on optical flow is used to calculate the homography matrix in the latter frames and when the number of inliers successfully tracked is less than 4, then it goes back to ORB feature point matching again. The result shows that the improved AndAR can augment not only reality based on markers but also reality based on planar natural features in near real time and the hybrid approach can not only improve speed but also extend the usable tracking range.

1. Introduction

Augmented Reality (AR) is an important branch of Virtual Reality (VR). It integrates virtual digital information into a 3D real environment in real time.

With the substantial increase in performance and penetration of smartphones, the researchers concentrate on Mobile Augmented Reality (MAR). The 3D registration technology is the main difficulty for AR or MAR and its performance directly affects the performance of an AR system. For MAR, the 3D registration is the tracking of the position and pose of the smartphone in the real scene in real time and so the virtual scene can be inserted seamlessly into the real world in real time using the information of position and pose. The 3D registration of MAR is mainly based on computer vision. Similar to AR, the 3D registration technology of MAR has undergone the change from markers based [1] to natural features based [2–4]. This change is determined by the improvement in the level of hardware and the demand for using AR outdoor. Currently, 3D registration technology based on natural features is a hot spot and the direction of development in the future. The natural features based 3D registration methods of MAR are derived from AR, but some optimizations must be done according to the characteristics of weak computing power of smartphones. In addition, since smartphones have GPS and various sensors which can provide information about the position and pose of cameras, it can not only achieve 3D registration based on GPS and sensors to create some MAR applications for navigation [5], but also combine computer vision with sensors to improve the speed and accuracy of 3D registration [6]. The latest trend is to combine MAR with cloud computing [7,8]. If it is developed, the application of MAR will be freer and more practical.

Of course, the quickest way to develop a MAR application is to use the AR Development Kit. Currently, the common AR Development Kit on the Android platform includes metaio AR, Vuforia, Dfusion and AndAR. This paper will introduce transplants from ARToolKit [9]. Though the 3D registration method of AndAR is still based on markers, the biggest advantage of AndAR is that it is open source compared to others. So AndAR is suitable for theoretical research on registration algorithm, improvement, and development of depth customization MAR application.

The remainder of paper is organized as follows. Section 2 analyzes the architecture and workflow of AndAR. The standard development process is shown in Section 3 and the improvement for AndAR is shown in Section 4. Section 5 presents the results of improved AndAR. Section 6 describes the future work.

☆ This paper has been recommended for acceptance by Luming Zhang.
* Corresponding author.
E-mail address: chenpeng@ctgu.edu.cn (P. Chen).

http://dx.doi.org/10.1016/j.jvcir.2015.06.016
1047-3203/

2. Architecture and workflow

AndAR is a project that enables AR on the Android platform and is an AR Framework for Android. It not only offers a pure Java API but is also object-oriented. The developers can find applications of MAR conveniently with AndAR.

2.1. Architecture

AndAR consists of three modules as shown in Fig. 1.

Camera Java API: image acquisition module. Class *Preview* extends from class *SurfaceView* and implements *SurfaceHodler*. Callback; Class *CameraPreviewHandler* implements interface *PreviewCallback*, input parameter *data* in *public void onPreviewFrame*(*byte*[] *data*, *Cameracamera*){} is the real time frame data which is needed in target detection and tracking module below. This module mainly deals with acquisition of real time image frame.

ARToolKit Java API: target detection and tracking module. Java Native Interface (JNI) is used to reflect related kernel functions of target detection and tracking in C language version ARToolKit into java functions. These functions are encapsulated in the Java class *ARToolkit*. This module mainly detects and recognizes markers and calculates transformation matrix needed by 3D registration.

OpenGL Java API: Render and display module. This module is based on OpenGL ES 1.0 graphics library and mainly completes rendering virtual scene in the real time frame according to the transformation matrix.AndAR is an open source software, and developers can revise it to satisfy their needs. WE will employ this advantage to improve AndAR later.

2.2. The workflow

One of the difficulties in developing AR or MAR applications is tracking the user's view point. AndAR uses computer vision algorithms to calculate camera's position and pose relative to physical markers, namely the camera extrinsic matrix in real time.

According to the camera imaging model, the imaging process can be described as the transformation process among the world coordinate, the camera coordinate and the image coordinate. As is shown in Eq. (1), in which K known as the camera intrinsic matrix is only concerned with the internal structure of the camera which can be determined by camera calibration and it is considered to be constant. R and t of matrix T_{cw} are the position and pose of the camera, known as the camera extrinsic matrix.

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u \\ 0 & \alpha & v \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{R} \quad \mathbf{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{KT}_{cw} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (1)$$

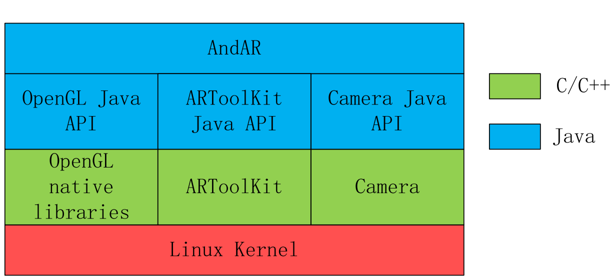

Fig. 1. Architecture of AndAR.

AndAR calculates the camera extrinsic matrix based on markers to complete the overlay of virtual imagery on the real world (3D registration). The advantage of the markers is that their shape and gray contrast sharply with the surrounding environment, so it is easy to be detected. The workflow of AndAR is designed based on the feature of the markers. Firstly, a color frame image is converted to a binary image according to a fixed gray threshold. Then connected components in the binary image are labeled and filtered as candidate areas according to the heuristic rules. Next, each candidate area matches with the standard templates in memory, so which marker being in the image can be known. The final step is to calculate the camera extrinsic matrix by the deformation of the marker's border and add the virtual scene to the developers can develop application the real time image frame according to the camera extrinsic matrix.

3. Standard development process

Whether the 3D registration is based on markers or natural features, the application runs on PC or mobile terminal, standard development process of a complete AR system includes the following steps.

(1) Initialize the camera and read in the relevant configuration files and the standard templates.
(2) Grab a frame from the real time video stream.
(3) Detect and recognize the target in the video frame image, and then calculate the camera extrinsic matrix.
(4) Render virtual scene and align it with markers or natural features by using the camera extrinsic matrix.
(5) Close the video stream, and disconnect the camera.

The process from 2–4 continues to cycle again and again. As mentioned previously, AndAR is an AR framework for Android and this framework achieves most of the development process above.

In order to create a simple AR application based on AndAR, as shown in Fig. 2, we need three steps according to the object-oriented programming.

(1) Create a class extended from abstract class *AndARActivity*, for example *CustomActivity*. *AndARActivity* is extended from *Activity* which is one of the basic modules of an Android application, generally on behalf of a screen of phone.
(2) Create a class extended from abstract class *ARObject*, for example *CustomObject*, and override the *draw* method. *ARObject* stands for the standard template whose construction method includes information about the standard template such as name and size and membership method *draw* defines virtual scenes, everything drawn in the method *draw* will be drawn directly onto the marker.
(3) Create an instance of *CustomObject* in *CustomActivity*, and call method *registerARObject* (*ARObject*) to add the standard templates and define the virtual scenes.

In addition, the standard template files and the camera intrinsic matrix file need to be put into file folder *assets*. When the application is running, the standard template files will be transferred to the private folder of the application in the smartphone so that the files can be read in by native C/C++ codes.

In this sample, the virtual model is just a cube in blue. AndAR project also provides a way to use *obj* models. This format can be exported by most 3D modeling software, for example, 3DS MAX. The data of 3D model in *obj* file should be loaded into application's memory firstly, and this might take a while. So it should be