Vision based First Perspective Shooting Game System with Smartphones

Sung Hongnyeom
Department of Software Engineering
P.O. Box 61186
Korea
171872@jnu.ac.kr

Hwang Heejae
Department of Computer Engineering
P.O. Box 61186
Korea
171413@jnu.ac.kr

Lee Chilwoo
Department of Computer Engineering
P.O. Box 61186
Korea
leecw@jnu.ac.kr

Daejin Kim
Department of Electric Engineering
P.O. Box 61186
Korea
djinkim@jnu.ac.kr

ABSTRACT

In this paper, we propose a new system that can realize a smartphone FPS (First Perspective Shooting) game in real time by improving the tracking error of the previously proposed game system [1]. This system is very useful for real-time play because it can synchronize information used by players while reducing tracking errors by memorizing important landmarks as feature points in the path traveled by multiple users.

KEYWORDS

Visual Odometry, Computer Vision, Location Synchronization

1 INTRODUCTION

FPS games that can be enjoyed outdoors by imitating actual battles can maximize realism and tension, so many tools and systems have been developed so far. A typical example is the survival game shown in Figure 1, where the victory or defeat is decided by eliminating the enemy using paint bullets instead of real bullets. However, these games have no choice but to be played under the supervision of a supervisor in a specially designated place in consideration of the safety and economic feasibility of the game. In the case of Figure 2, expensive equipment like laser sensor is sometimes required because killing an enemy is determined by the exact location of the strike. This fact makes it impossible to simply enjoy this game anytime and anywhere.

Figure 1. A survival game using paint gun.

Figure 2: Group FPG using laser tag device.

To solve this problem, we proposed a system for enjoying FPS survival games using smartphones in a previous paper [1]. This system calculates *Visual Odometry* in a mobile environment to track gamers' movements in real time and demonstrating that games are possible. However, as shown in Figure 3, there was a problem in that a lot of errors were accumulated in the process of calculating Visual Odometry. In addition, the problem of synchronizing the positions of multiple players was also problematic in that it was impossible to accurately synchronize the position information depending on the initial position and direction information obtained through GPS.

To solve these problems, this paper proposes a system that can precisely track and synchronize the location of multiple clients based on the algorithm proposed in the paper [2].

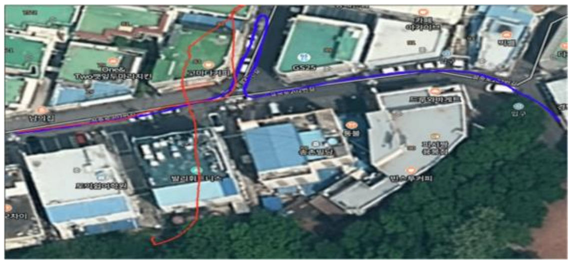

Figure 3: Error accumulation occurred in previous research[1].

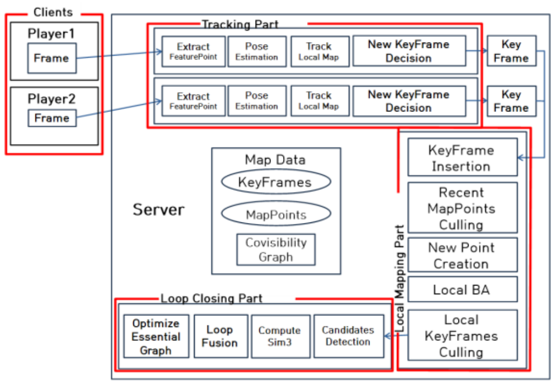

Figure 4: System Configuration

2 Real-time location estimation and synchronization for multiplayer

The overall configuration of the system proposed in this paper is shown in Figure 4. This configuration has been improved so that the method proposed in the paper [2] can be used in the real world, and actual location estimation and synchronization of multiple players becomes possible.

The proposed system is largely composed of five parts: the client part user's mobile device, the tracking, local mapping, and local closing parts, which are calculation parts, and the map data part, which is a data management part. In this paper, unlike the previous paper [1], for location synchronization, the information used to calculate the location of each mobile client is stored in the server under the name of “MapData”, and this information is shared by the clients. And in the calculation part, based on the camera frame of each client, real-time location calculation in the tracking part, MapData management in the local mapping part, and finally MapData correct in the loop closing part.

2.1 MapData

In the system proposed in this paper, we use MapData to synchronize the positions of players in real time along with the calculation of Visual Odometry of each client; a smartphone. MapData is information about important feature points, namely landmarks, that are selectively used in the process of calculating the movements of players using Visual Odometry. MapData consists of Map Point, Keyframe, and the correspondence between Keyframes. Each MapData component is as follows. First, the keyframe is a frame that can be characterized in the process of creating a map among the frames received as input. Keyframe is created and managed in the same way as Table 1 below.

Each Key Frame constructed through this process has Feature Points. These Feature Points are projected on the 3D space and stored in Map Data under the name of MapPoint. Lastly, the Covisibility Graph shows the relationship between Key Frames in the form of a graph. In this representation, nodes represent Key Frames, and connecting lines between nodes represent Map Points shared between Key Frames.

Table 1: MapData Management process

1. Continuous video frame input to Tracking Part
2. Select a frame that is not too similar to the last selected Key Frame and has more than 50 Feature Points as Key Frame candidates
3. Optimize after comparing the selected Key Frame candidates with Key Frame of MapData

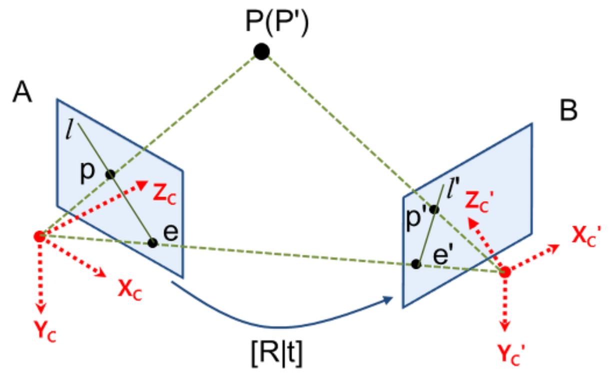

Figure 5: Epipolar Geometry constraints.

2.2 Tracking

In the system configuration diagram, the Tracking method estimates the current location by using the Feature Point of the Frame input from the client and MapData stored in the past. In this case, in the previous paper [1], the location of the corresponding feature point between the previous frame and the current frame was estimated using the Epipolar Geometry constraint as shown in Figure 5. In this case, the Odometry calculation method is shown in Table 2. However, in the system of Figure 4 newly proposed in this paper, MapData information is used to estimate the current location in addition to the previous frame. Therefore, unlike previous papers, in addition to the previous frame, Key Frames are also used to obtain the Essential Matrix, enabling accurate Odometry calculations. This method can also solve the position synchronization problem, which is another problem in the previous paper. In the previous method, only the previous frame information of the device was used, so information could not be shared with other users, resulting in a synchronization problem. However, in the new system, each client allocates one method to estimate the location using the MapData information shared between clients as well as the Feature Point of the previous frame, so that the relative location can be synchronized in real time.

Table 2: Visual Odometry calculating process

1. When the number of remaining feature points is less than 2000, search for feature points by using the FAST algorithm.
2. Using KLT Tracker to match the feature points of the previous frame to the current frame
3. After calculating the five most correctly matched feature point pairs through RANSAC, the essential matrix is estimated with the 5-Point algorithm
4. Calculate the trajectory between frames by finding the rotation matrix and translation vector in Essential Matrix

2.3 Local Mapping

The Local Mapping method analyzes and manages the information between the Key Frame selected in the Tracking method and the existing MapData. The Local Mapping method receives as an input the Key Frame selected according to the criteria in Table 1 in the Tracking method. Then, the feature point correspondence between the corresponding Key Frame, the existing Key Frame of Mapdata, and the Map Point is analyzed. After that, based on the analysis result, the correspondence between Key Frame and Map Point as shown in Figure 5 is stored in the Covisibility Graph. Then, based on the information, new feature points are triangulated, and the three-dimensional position is stored inside the map. Finally, the MapData is optimized by removing the Keyframe and Map Point, which have weak correspondence. Finally, the relative position of each keyframe is corrected based on the covisible graph.
In Figure 6, the dots represent the Map Point, the rectangle means the location of the Key Frame, and Figure 7 shows the relationship between information in MapData.

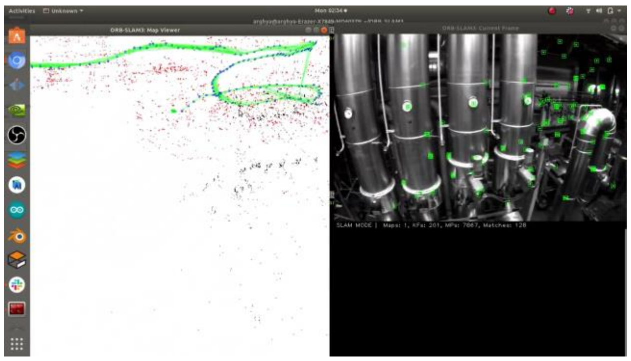

Figure 6: Visualization of Mapdata

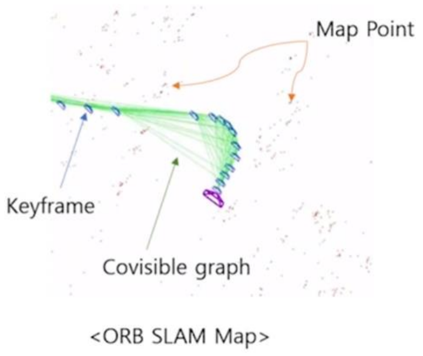

Figure 7: Mapdata composition and interrelationships.

2.4 Loop Closing

Finally, this part is a method that can correct drift that may occur during the position estimation process. In this method, if it is determined that the client has returned to the previously visited place based on the correspondence of MapData, it is judged that the client has returned to the same location and the error occurring between them is corrected as shown in Figures 8 and 9.

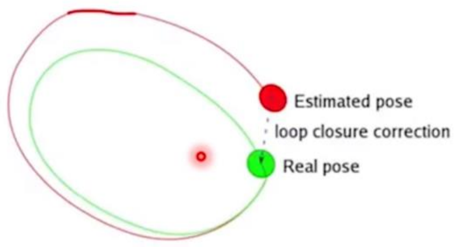

Figure 8: Mapdata composition and interrelationships.

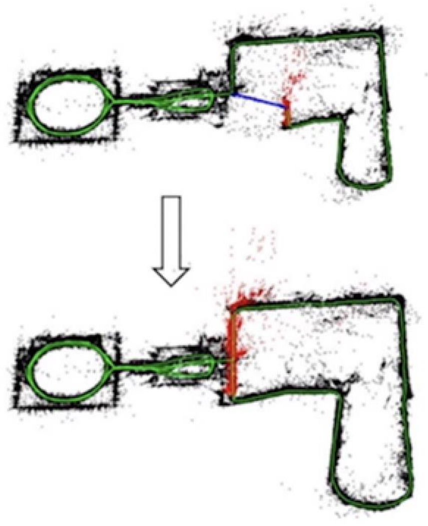

Figure 9: Correction result of actual error.

3 CONCLUSION

In this system, we describe the newly designed a FPS game system to correct the errors that occurred in the previous paper [1] and to synchronize the information used by multiple players. This system composes and uses a kind of point cloud by composing a map for the space the players have moved through MapData in the odometry estimation process. And since each client creates a Map while moving together in the same space, a more sophisticated Map can be obtained. Also, since all clients share the MapData configured in this way, it will be possible to synchronize motion information naturally without sharing direction and location information with GPS or magnetic field sensors.

ACKNOWLEDGMENTS

This research is supported by Ministry of Culture, Sports, and Tourism(MCST) and Korea Creative Content Agency(KOCCA) in the Culture Technology(CT) Research & Development Program(R2020060002) 2020.

REFERENCES

[1] Hongnyeom Sung, et al., “Computer Vision-Based AR Shooting Game”, The 18th International Conference on Multimedia Information Technology and Applications, 2022

[2] CAMPOS, Carlos, et al., “Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam”, IEEE Transactions on Robotics, Vol. 37.6, pp. 1874-1890. 2021