CS 491 Senior Design Project

Specification Report

DepthCube

Supervisor:
Prof. Uğur Güdükbay

Students:
Alican Büyükçakır
Serkan Demirci
Semih Günel
Burak Savlu

Jury:
Assoc. Prof. Selim Aksoy
Prof. Özgür Ulusoy

Innovation Expert:
Armağan Yavuz (TaleWorlds)

October 10, 2016

Contents

Introduction

1.1 Description

DepthCube is a portable game console that extends mobile devices' environmental understanding for augmented reality (AR) purposes. DepthCube takes your everyday environment and turns it into a virtual playground which you see through mobile device's camera. We want to create an environment where virtual objects interacting with the real formation of your environment. In this sense, players will interact with the game console using their mobile devices as gamepads.

In the core application, DepthCube will provide device/application with the precise location of the device relative to the room with the fine details about formation of the environment. We want to achieve this goal using state-of-the-art computer vision algorithms, with minimizing the need of an additional hardware. The problem with the current state of the AR applications is the lack of computational power on the mobile devices, combined with the difficulty of providing real-time environmental formation. We want to tackle this obstacle with the additional computing power provided by DepthCube and better algorithms.

Our ultimate goal is to create an API-like infrastructure, where many interesting ideas can be built upon. We can see many innovations originated from the Kinect console, which provided developers with the depth sensor and skeletal tracking. It is possible to do the same thing with the minimum cost, but this time on the mobile devices, with a new idea. Customers will be willing to buy DepthCube in order to extend their mobile devices capabilities on the environmental understanding and enable potential applications exploiting this area awareness.

To further clarify, assume that we have a game that is already developed and ready to play for the users developed using the API of DepthCube. Our exemplary new game will be very similar to Pong, but this time balls bounce on the real walls and objects, rather than on rectangular borders. Users will interact with the virtual ball using their mobile device as a virtual surface where virtual ball bounce on. Here is how the system will work:

- First user will scan her environment using the camera present on DepthCube. Simultaneously DepthCube will create a sparse representation of the environment, and show the 3D model on the mobile devices. It is sufficient to scan a room only once, when the console is newly acquired.
- When scanning phase has completed, DepthCube will create a dense representation of the room in the offline mode. This construction will take several hours. At the end DepthCube will create the environment in Unity model, which correctly represents the room.

Figure 1.1: DepthCube's understanding the room after scan phase. Each vertex corresponds to a 3D point.

- Then, Unity model will be transferred to the mobile device which runs the DepthCube Android application, which is built upon Unity Engine.
- Once Unity model is received and wireless connection between the mobile device and DepthCube is console is established, users can run the Pong game on their phones.
- During the game phase, wireless connection between DepthCube and device will provide the information of precise localization of mobile device.The information will be extracted from key-points, intensity information using already calculated dense 3D structure of the room.
- Since Unity engine has the fine details of the room with the precise location of the mobile device, Unity engine will draw the virtual objects in the correct location and size without disrupting the perspective.

1.2 Similar Products / Technologies

There are some products that use similar technologies to the DepthCube. There are game consoles such as Kinect, or additional sensors like Structure Sensor use depth information to build 3D structure of the environment. Although, they can perform camera tracking and 3D model construction faster, these sensors have limited range for their depth measurements. They cannot construct 3D model of the large environments. Moreover, their price is drastically more expensive than DepthCube's targeted price. Additionally, there are some AR frameworks that run for mobile devices. With these frameworks, developers can build AR applications that can interact with the environment without need of any external hardware. However, they cannot perform well due to low computational power of the target mobile devices. In addition, they use a lot of battery for their calculation.

The most similar technology with DepthCube is Project Tango, currently developed by Google. Project Tango wants to solve the problem of area awareness in real time, and their ultimate goal is the simultaneous localization and mapping. Tango devices include a powerful NVIDIA GPU, a depth sensor, and RGB camera with large field of view. All these to support real-time computations. However these which makes the Tango device relatively expensive, compared to main-stream mobile devices, and it is almost stands like specifically built for AR and VR purposes, rather than serve as a general mobile device. Tango device is heavy and thick compared to the main-stream phones and tablets.

1.3 Constraints

1.3.1 Technical Constraints

- Raspberry-Pi will be used as the backbone of our gaming console. It will be responsible for taking care of the computationally heavy part of the system, as well as storing scenery data.

- The wireless card of the Raspberry-Pi will be utilized for communicating between mobile devices. Raspberry-Pi will also act as a server in this manner.

- A monocular camera that will be mounted to the Raspberry-Pi will be used for scanning the environment.

- Mobile device on which the augmented reality experience will be held needs to have a back camera.

- The AR applications will be available on Android operating system.

1.3.2 Social Constraints

- The developer framework will be open-source, allowing everyone to fork, develop their new ideas or improve existing ones.

- Multiplayer option of the developed games in the gaming console will help people to build social relations.

1.3.3 Implementation Constraints

- The structure of environment will be turned into game scene using Unity3D.

- Game developers will use Unity3D to develop apps for DepthCube.

- Network between mobile devices and console will be established using TCP/IP protocol.

- User's security details will be stored in a MySQL database.

- GitHub will be used as the version control repository.

1.3.4 Sustainability Constraints

- Our gaming console should have low energy consumption.
- Distributing the computationally-heavy part of the process to our gaming console will decrease the power consumption of the mobile devices, therefore increasing their battery life.

1.3.5 Economic Constraints

- The cost of the gaming console will not exceed $100.
- The framework that will be developed will be under GNU General Public License 2.0 and open-source. It will not require any payment for other developers to use.
- RGB-D sensor or stereo camera will not be used in acquiring scene data, due to their high prices increasing the overall cost of the product.
- Powerful GPUs that would normally increase the computation capabilities of the console will not be used, due to its high price.

1.3.6 Security Constraints

- Any kind of user data such as usernames, emails or passwords that is required by the application that uses our framework will be encrypted and secure.
- The latest 3D models that the user created will be encrypted and secure.
- Secure connection will be established between the console and the mobile devices, to prevent any information leak.

1.3.7 Language Constraints

- The console will be used worldwide hence it will have English as the default language.
- Other languages may be added depending on the console's popularity.

1.4 Professional and Ethical Issues

The secrecy of the users will be of the utmost importance. As users share their information of their private rooms, we are obligated to protect what is already theirs. We want to achieve this goal firstly using a username/password pair. Since we will already support multiple users, this will be important to prevent each user from accessing the information of the other corresponding users.

One other problem is with the possible danger of reverse-engineering, where adversary can access the memory after opening the case of the DepthCube. If we decide to leave the structure information in the game console without any protection, it can be easily decoded, bypassing the need for a username/password pair. To prevent this we need to encrypt the room information and decrypt it only when username/password is correctly identified. Once the game session has ended, decrypted information should be deleted until a new session.

On the other hand a professional problem is the usage of already licensed software. Since our design will heavily rely on the state-of-the-art computer vision algorithms, we have to use already existent software, since writing it from scratch is virtually impossible given the complexity of the algorithms. However since each software relies on many dependencies, we should be cautious of the possible violations of the licenses.

We will take the best available security measurements in order to protect user's secret about their rooms. Formation and location of the rooms of users should be strictly private, and DepthCube should be unalterable for any possible adversarial use. At the end of the day no adversary should be able to exploit the trust users have placed in DepthCube.

Requirements

2.1 Functional Requirements

- Players should have their own unique username/password.
- DepthCube should not require anything other than an Android device to run.
- Games should be controlled using a mobile device(smartphone/tablet).
- There should be two types of users for the product: developers and players.
- Players should be able to experience the Augmented Reality, given an already developed and ready-to-play game.
- Players should be able to upload videos for 3D construction.
- DepthCube should process user videos to create 3D structure
- Players should be able to download games from the online library.
- Developers should be able to design and develop their own games using our API that provides the location and structural information about the environment.

2.2 Non-Functional Requirements

- Each player should have access to the last 5 3D structures they created.
- DepthCube should be pocket-size.
- DepthCube should support Android 4.4 and above.
- A local and fast connection between DepthCube and mobile device is required. The communication latency between the DepthCube and the mobile device should be less than 40 ms.
- DepthCube should not require any kind of internet connection to work.
- Mobile interface of the DepthCube application should be user-friendly, clean and simplistic. Users must not have any problems with navigating themselves inside the application.
- Games developed for DepthCube should have at least 30 FPS.

- The provided API for developers should be well-documented, precise and succinct.
- DepthCube should scan user video and create a 3D structure in less than 1 hour.
- The provided API should be developed on Unity3D.
- DepthCube should support games developed using Unity3D version 5.0.0 and above.
- The server code should be written in C++.

References

Project Tango:
http://get.google.com/tango/

Structure From Motion:
http://bit.ly/2dSxD7L

OpenCV Camera Motion Estimation:
http://docs.opencv.org/3.1.0/d5/dab/tutorial_sfm_trajectory_estimation.html

CS491 Senior Design Project I - Guidelines:
http://www.cs.bilkent.edu.tr/CS491-2/CS491.html

ORB-SLAM, structure from motion implementation for monocular cameras:
http://webdiis.unizar.es/~raulmur/MurMontielTardosTRO15.pdf

Kinect with Skeletal Tracking:
https://msdn.microsoft.com/en-us/library/jj131025.aspx

Kinect Sensor:
https://msdn.microsoft.com/en-us/library/hh438998.aspx

Kinect Sensor Hardware Specifications:
https://developer.microsoft.com/en-us/windows/kinect/hardware

Structure Sensor:
https://store.structure.io/store