Digital Interactive Game Interface Table Apps for iPad

Supervised by:   Professor Michael R. Lyu
Student:         Ng Ka Hung   (1009615714)
                 Chan Hing Faat   (1009618344)

Year 2011 – 2012 Final Year Project

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Term 1 review

New Ideas, new target

Our work

Implementation

Conclusion
What is AR?

Combination of reality + Computer generated graphics

Interactive & digitally manipulable
Digi.T.able (2007)

- Allows players in different places to play games by real objects
- Share a same common space

Implement on iPad
i.Digi.T.able

2 iPAD shares a common AR space to play a game
Objectives

**i.Digi.T.able**

- Track the real-object mark and determine the camera’s position
- Display simple objects on virtual space depends on real space scenes
- Exchange position information between 2 iPad clients
- Implement a simple AR game on iOS platform (iPad)
Development tools

Objective-C on Xcode
Development tools

- PHP
- SQLite
A demo game

Dodge ball

- 2-player battle version

SETUP

- A marker on the wall
- 2 iPads with app installed
- Server ready
The game

A demo game

Dodge ball

Control
- Move around the device to move
- Tap to throw a ball
The game

Admin view on web interface

DEMO
The game

Limitations

- Unstable network
- Uncomfortable control
- Not enough AR effect demonstrated
New Idea, new target
Game - Pong
Design

Arena

Paddle

Wall

Ball

Paddle

Wall
Focus

- Better control
- Better networking support
- Improved AR experience
Game options

- Single game (with AI bot)

Level of difficulties

Easy
Normal
Hard
Game options

- Online game (with user opponent)

  Connect

  Choose as player 1 / 2

  Play online
Pong

Single + multiplayer game

DEMO

iDigitable

i.Digi.T.able
Implementation

4 main components

- Camera input
- iOS Application
- Virtual world construction module
- Game Engine
- Marker tracking module
- Network connection module
- To screen output
- HTTP
Implementation

Marker tracking

Qualcomm AR SDK (Vuforia)

- fetches live streaming from the device camera
- The platform consists of these components:
  - Camera
  - Image converter
  - Tracker
  - Renderer
  - Application Code
  - Target Resources
Implementation

Marker tracking

**Vuforia**
- Trackable Markers

- right-handed coordinate system is used
Implementation

Marker tracking

**Control**
- Move iPad
  - Paddle moves

Relative positioning  vs  absolute positioning
Network

Connection protocol

Register phase
- gets a token
- use the token to connect again
(due to HTTP’s stateless property)
Network

Server

Database based

- more efficient
- easier implementation
Network

Communication

JSON (JavaScript Object Notation)

- standard communication format
- many library support
- High writability
- High readability

```json
{
  "clientToken": "btfpm7d3qj7pagirfarvur64b5lk56",
  "eventId": "2",
  "eventType": 3,
  "coordination": {
    "x": 20,
    "y": 30,
    "z": 0
  }
}
```
Design

Network

Communication

JSON

Event Object
- Coordination
- Event code
- Etc.

encode

JSON string

Transmit over internet

decode

JSON string
Game engine

Architecture

Core

The **logic part** is the major part of the system. All game logic related objects are involved in this part.

The **presentation part** is highly connected with OpenGL library - about how to draw things on the screen.
Design

Game engine

Logic part

- **Game**: represents a Pong game.
- **Player**: represents a game player
- **Bot**: represents a computer controlled entity
- **Paddle**: represents the rectangular block for hitting the ball
- **Ball**: represents a block that bounces between players
- **Court**: represents the game arena
- **Motion**: represents the ball’s motion states
- **Contact**: represents the contact point made by the ball and paddles
Most of the game objects such as the paddles, the ball and the arena need to be shown on the screen. Hence, they are associated with models for presentation.

typedef struct _Model : Object {
    int parentId;
    bool hidden;
    QCAR::Vec3F position;
    QCAR::Vec3F scale;
    QCAR::Matrix44F transform;
} Model;
To summarize this semester

- Improvement on tracking AR marker
- Updated QCAR SDK
- Network modification
- Pong game
Conclusion

Overall in 2011-2012

- Track the real-object marker -> determine the camera’s position
- Display simple objects on virtual space depends on real space scenes
- Exchange position information between 2 iPad clients
- Implemented a simple AR game on iOS platform (iPad)
Evaluation

Issues...

- Programming on iOS
- Searching for suitable SDK
- Stabilize camera tracking
- Network Connection
- Investigate possibility for more clients
Q&A Section
Thank you!