i.Digi.T.able

LYU 1103

# 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

# Agenda

Introduction & background

Project Idea

Our work

Conclusion

# AR?

Augmented Reality

# What is AR?

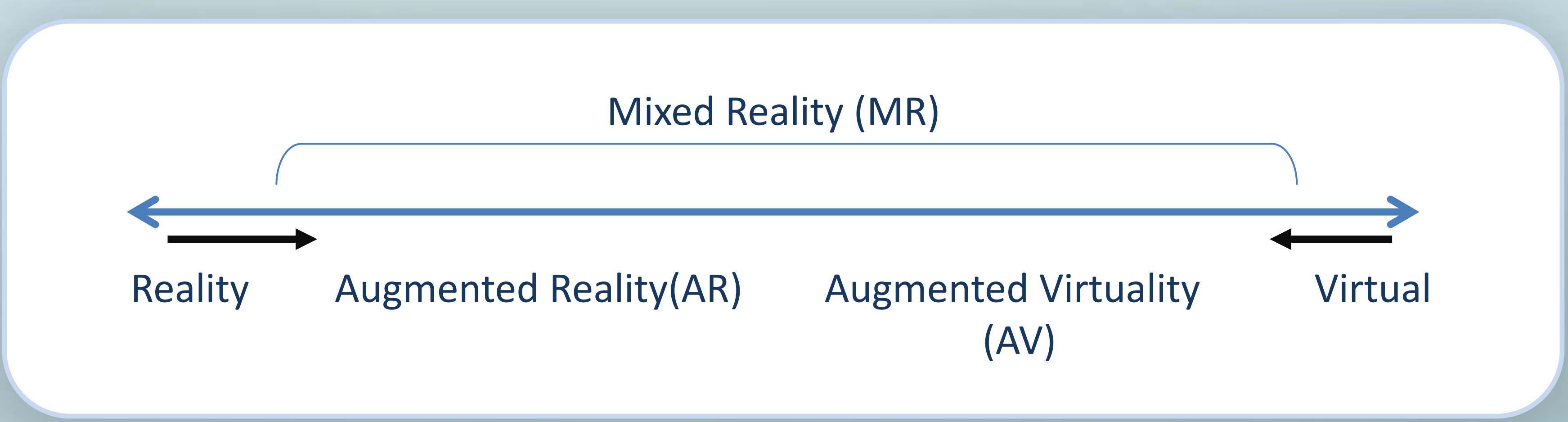
Combination of reality

+

Computer generated graphics

Interactive & digitally manipulable

# What is AR?



Mixed Reality (MR)

Reality

Augmented Reality(AR)

Augmented Virtuality (AV)

Virtual

**Enhance** realism and impressiveness

# AR types

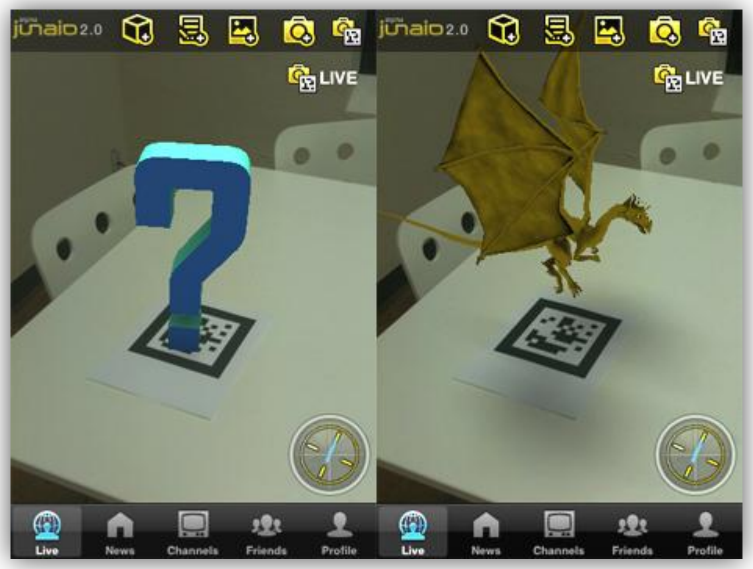
## Marker-less

- GPS
- Digital compass
- Camera assisted

# AR types

## Marker-based

- Camera
- Analyze marker
- e.g. QR code

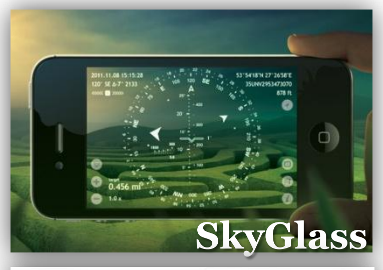


# AR examples

## Applications

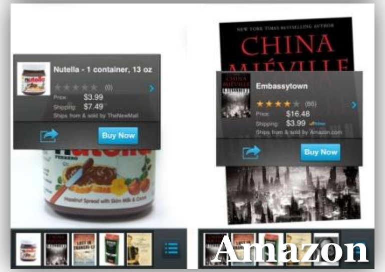
- **Geo-navigation**

  Compass, etc.



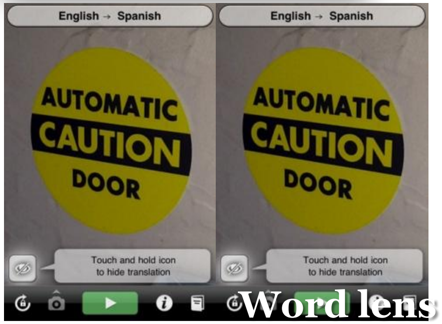
- **Informative**

  Stores



-**Translation**

  Direct view



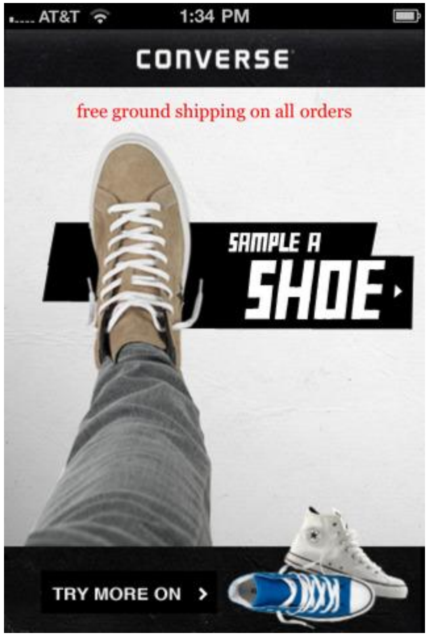
# AR examples

## Applications

- **Samplers**

As an interactive prototype

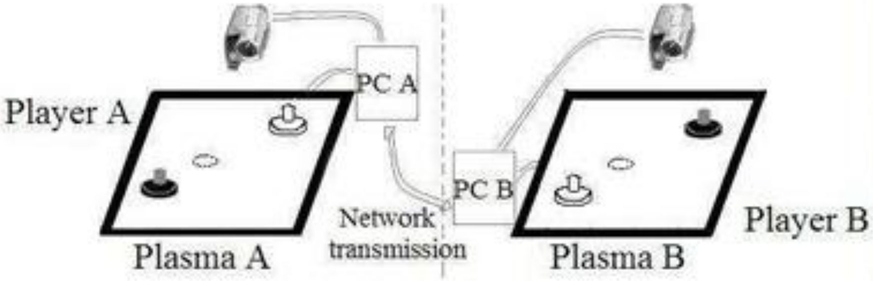
Product Advertisement



# Background

## Digi.T.able

A project supervised by Prof. Michael Lyu in 2007



<- Over-head Video Camera
Local Display
Remote Display
Tele-Table: VIEWLab, CSE, CUHK 2007

# Background

## Digi.T.able

- a multi-purpose interactive table

- allows players in different places to play games by real objects

- Board games (e.g. Chinese chess, uno)

- Action games (e.g. snooker, air hockey)

# Inspiration

## Digi.T.able

allows players in different places to play games by real objects

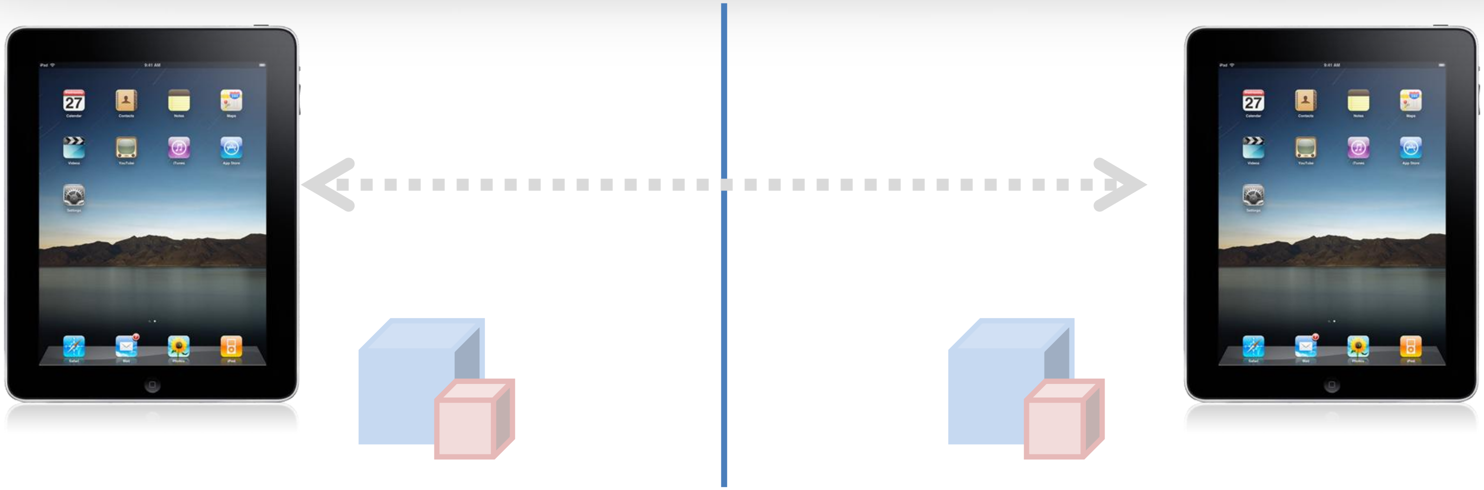
Share a same common space

**Implementation** on iPAD

# Idea

## i.Digi.T.able

2 iPAD shares a common AR space to play a game



# Objectives

## i.Digi.T.able

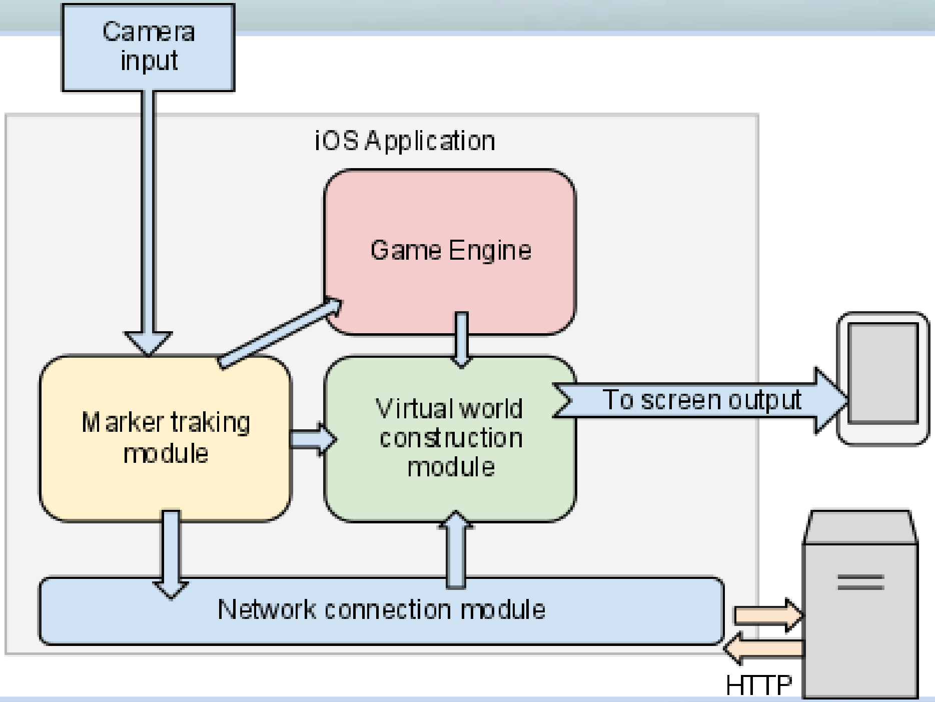
- Track the real-object mark and determine the camera’s position
- Display simple objects on virtual space depends on real space scenes
- Exchange position information between 2 iPad clients
- Implement a simple AR game on iOS platform (iPad)

# How we do that

i.Digi.T.able

## Design

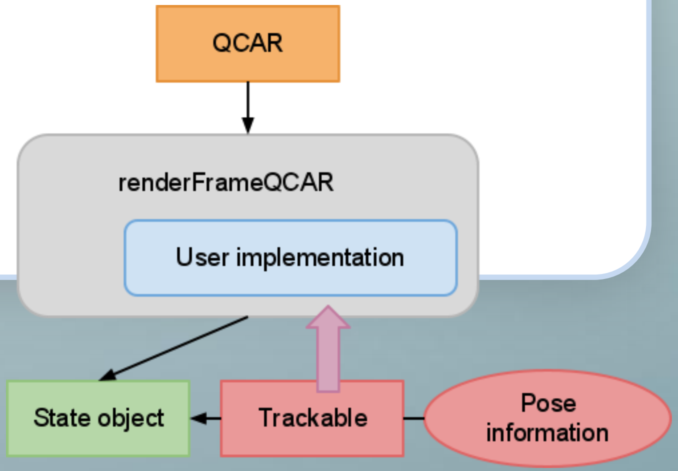
**4 main components**



# Design

## Marker tracking

The Marker Tracking Module is the agent that directly communicate with the Qualcomm AR SDK. It is a finite state machine keep analyzing data from camera.



# Design

## Marker tracking

### Detection and recognition

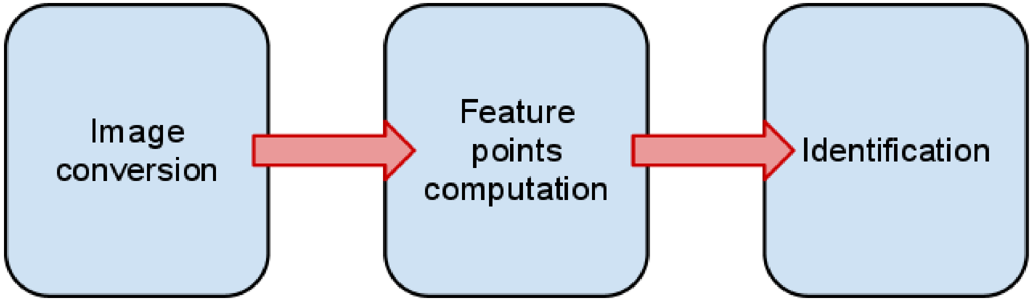


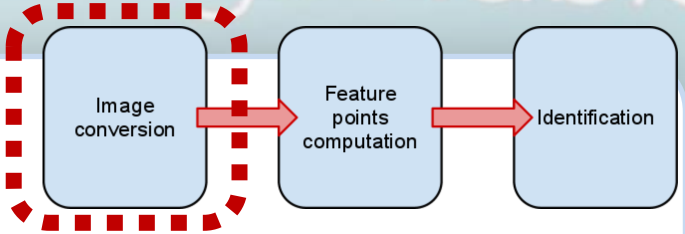
Image conversion

Feature points computation

Identification

# Design

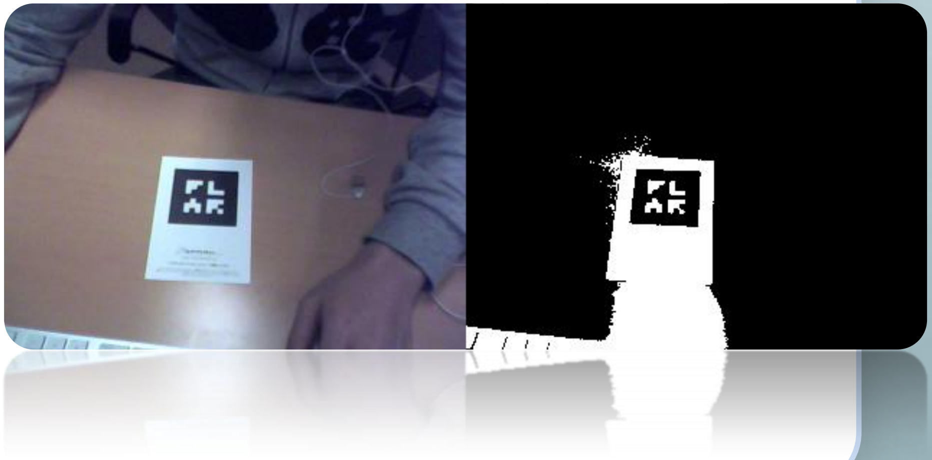
## Marker tracking



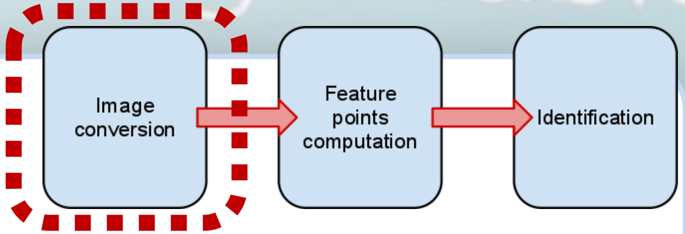
### Image conversion

- thresholding

- convert the captured frame from colored into binary image



# Design

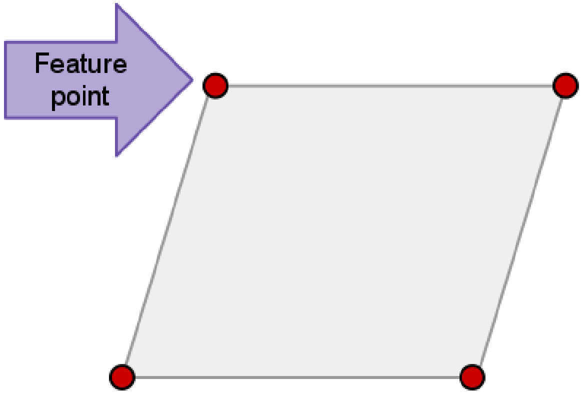


## Marker tracking

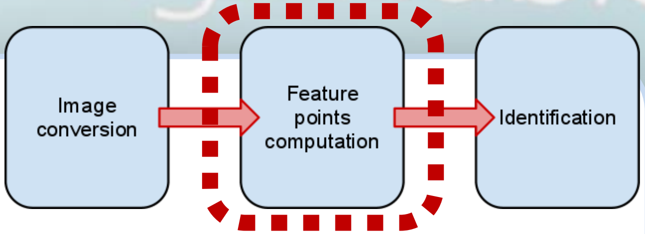
### Feature points computation

- corners need to be detected

- in order to have reliable camera pose estimation



# Design



## Marker tracking

### Identification

- restore the effect of **rotation, translation and perspective transformation**

by solving a simple linear system

# Design

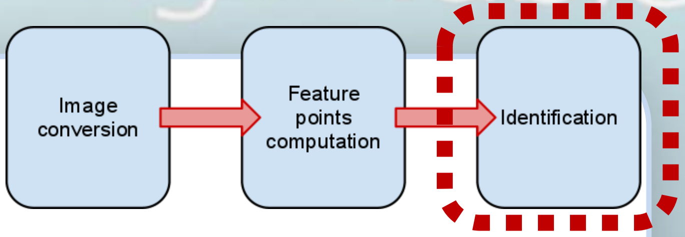


Image conversion

Feature points computation

Identification

## Marker tracking

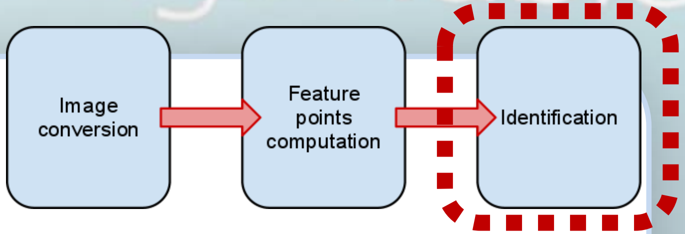
### Identification

- the positions of four corners by feature points computation and the 3D coordinates in object space of the marker's corners are given by (xi, yi, 0)

$$X_i = \frac{a_1 x_i + a_2 y_i + a_3}{a_7 x_i + a_8 y_i + 1}$$

$$Y_i = \frac{a_4 x_i + a_5 y_i + a_6}{a_7 x_i + a_8 y_i + 1}$$

# Design



## Marker tracking

### Identification

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \\ X_8 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -X_1 x_1 & -X_1 y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -X_2 x_2 & -X_2 y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -X_3 x_3 & -X_3 y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -X_4 x_4 & -X_4 y_4 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -Y_1 x_1 & -Y_1 y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -Y_2 x_2 & -Y_2 y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -Y_3 x_3 & -Y_3 y_3 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -Y_4 x_4 & -Y_4 y_4 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{pmatrix}$$

The result vector of the linear system implies a normalized marker. By using the result vector, the system can provide users the pose information for drawing virtual 3D objects.

# Design

QDevNet™
Qualcomm Developer Network

## Marker tracking

### Qualcomm AR SDK

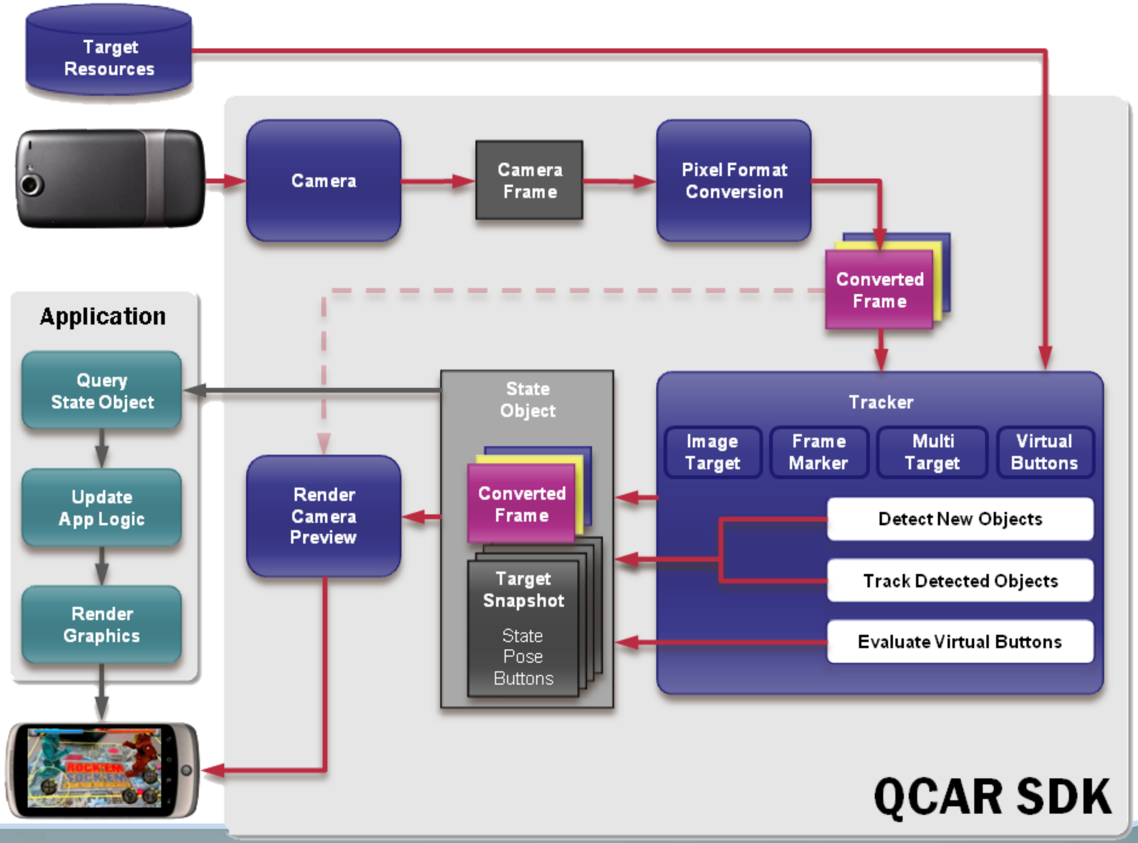
-fetches live streaming from the device camera

- The platform consists of these components:

  -Camera
  -Image converter
  -Tracker
  -Renderer
  -Application Code
  -Target Resources

# Design

## Marker tracking



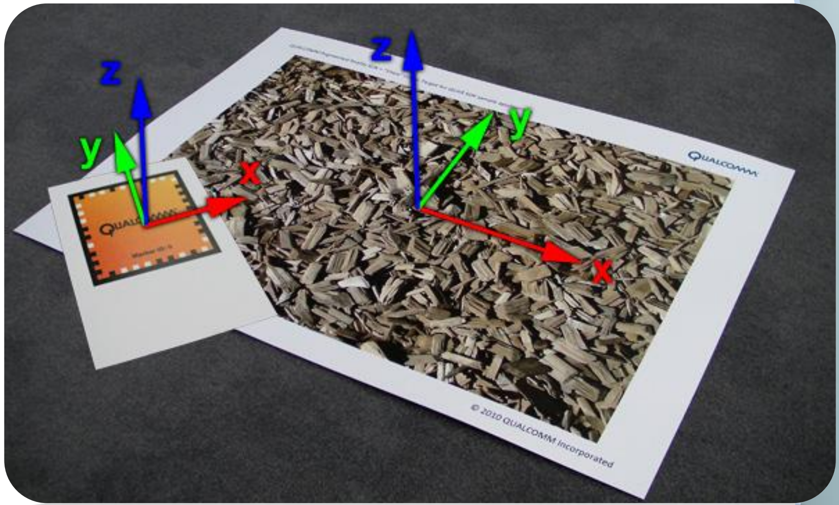
# Design

## Marker tracking

Qualcomm AR SDK

-Trackables

-right-handed coordinate system is used



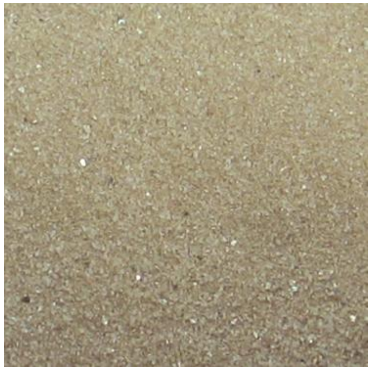
# Design

## Marker tracking

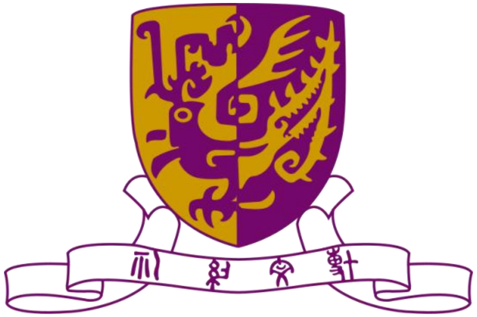
How to choose input images?

- Rich in detail
- Good in contrast
- No repeatitive patterns

### Examples



Not good

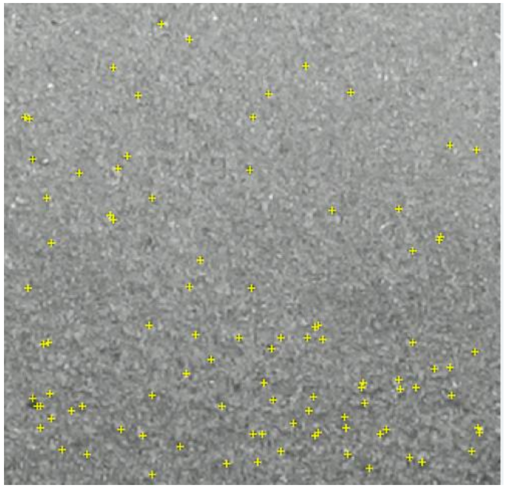


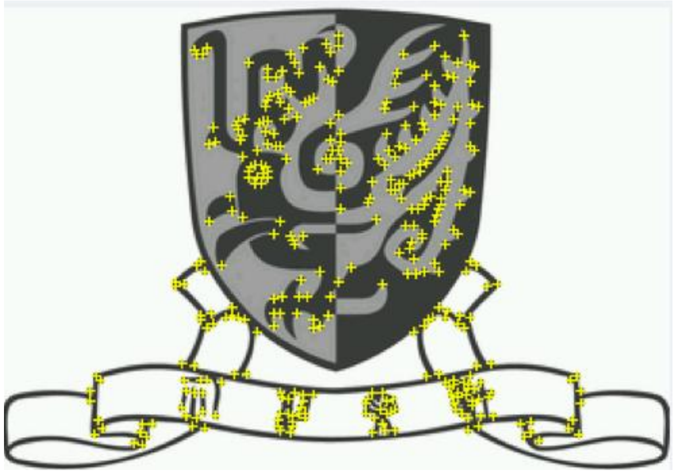
Good

# Design

## Marker tracking

### Examples





# Design

## Marker tracking

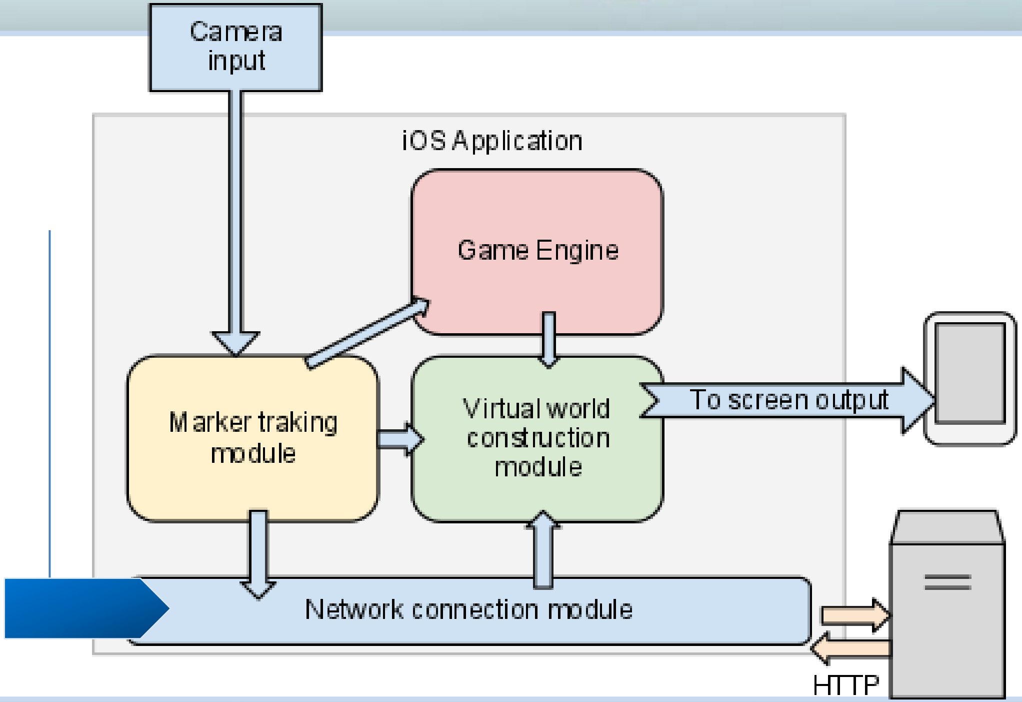
### **Qualcomm AR SDK** Compare with **String AR**

| | **Qualcomm AR** | **String AR** |
|---|---|---|
| **License** | Free | Free for limited version |
| **Platform** | iOS, Android | iOS, Android (in progress) |
| **Multiple markers** | Yes | No |
| **3-rd Party Integration** | Yes, Unity3D | Yes, Unity3D |

# Recap

## Design

**4 main components**



# Design

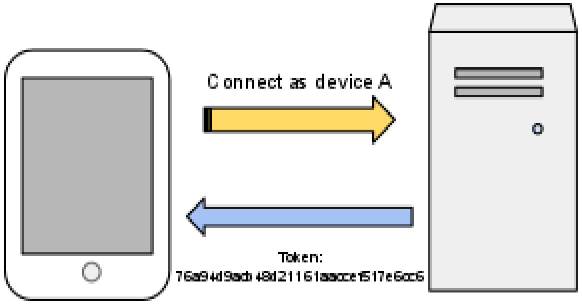
## Network

### How to exchange data efficiently?

- Data size , data type

-Network load, frequency of update

-Accessibility

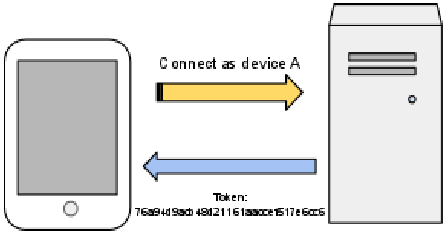


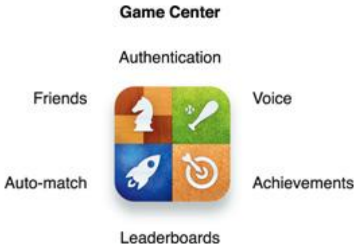
# Design

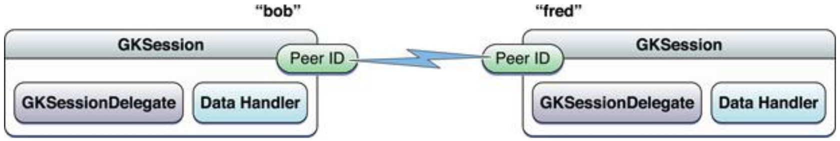
## Network

### Consider…

- Network socket
- HTTP requests
- Game center
- Peer-to-peer







# Design

## Network

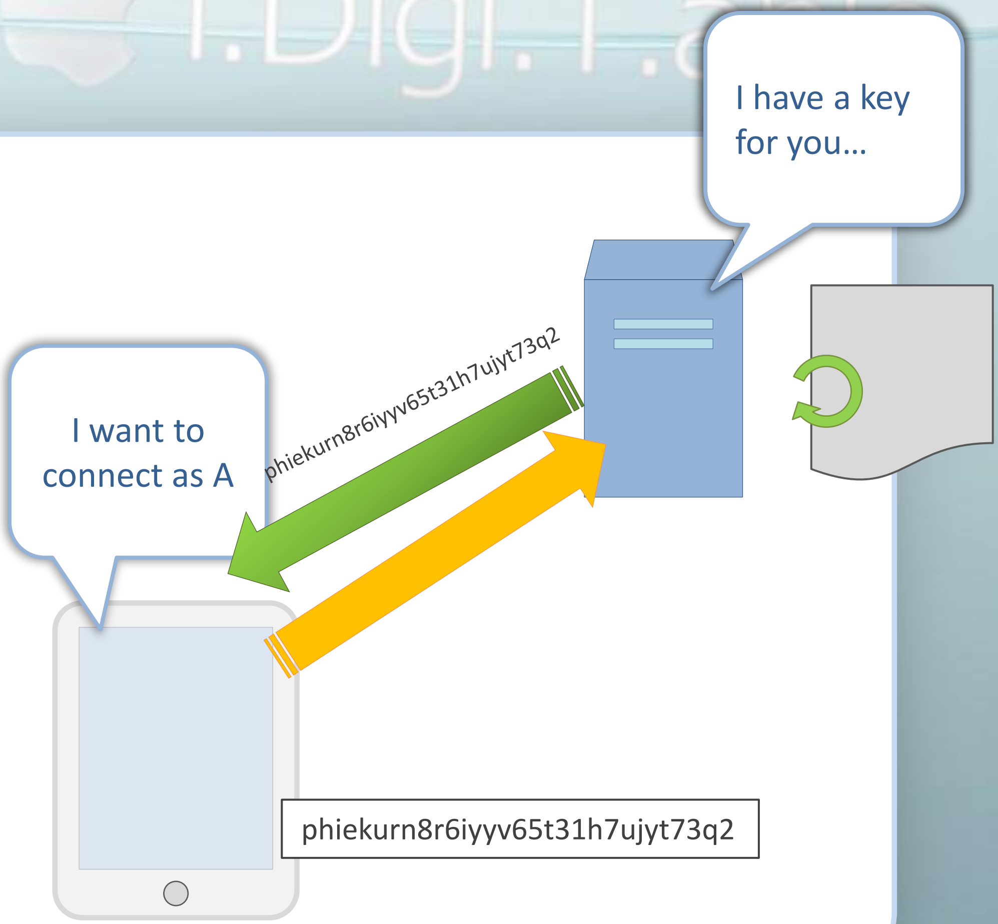
### Connection protocol

**Register phase**

- gets a token

- use the token to connect again

(due to HTTP's stateless property)


I have a key for you...
I want to connect as A
phiekurn8r6iyyv65t31h7ujyt73q2
phiekurn8r6iyyv65t31h7ujyt73q2

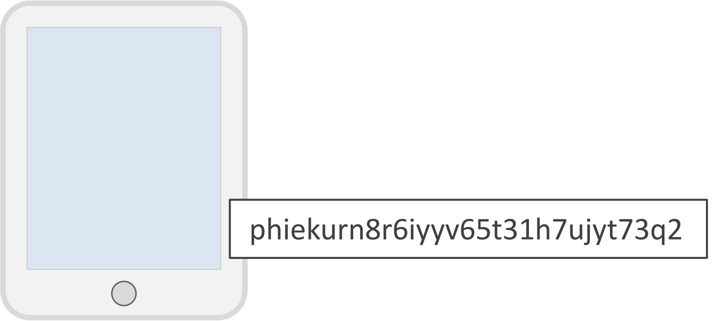
# Design

## Network

### Connection protocol

**Register phase**

- No IP involved
- Unique device identification



# Design

## Network

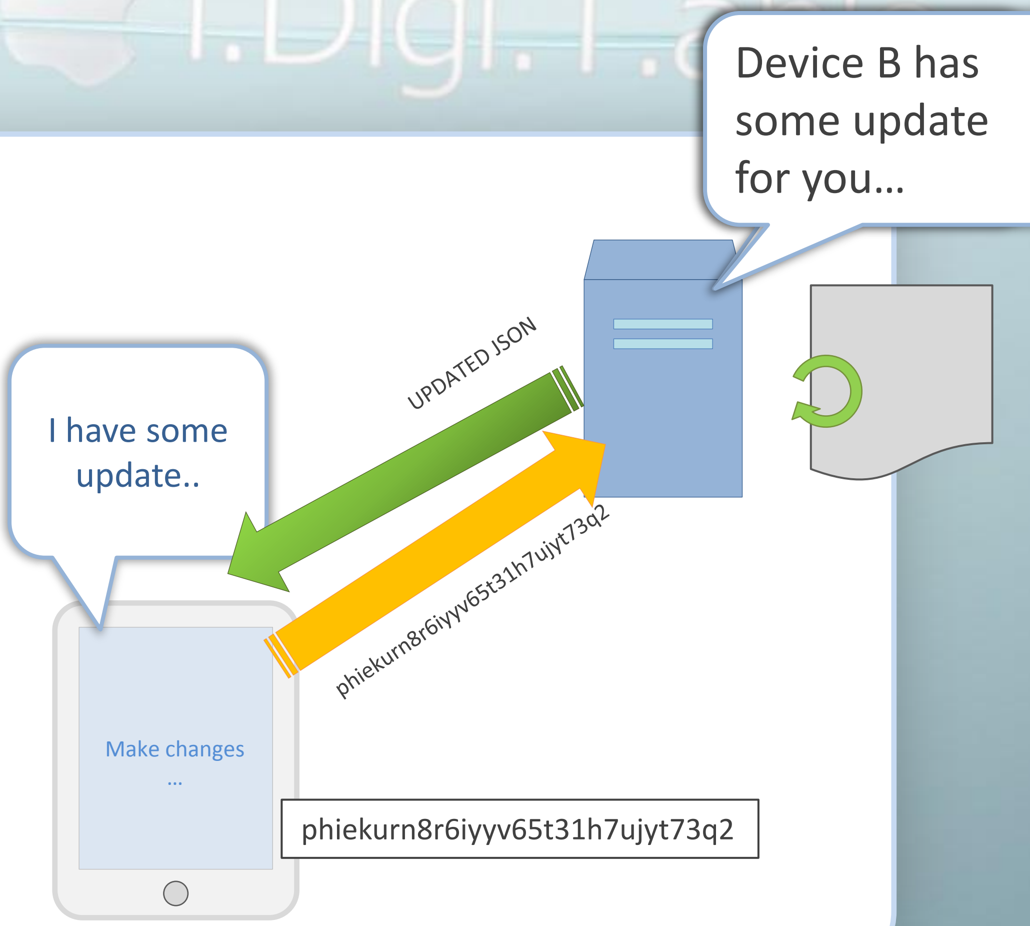
### Connection protocol

**Data update phase**

- present token

- update information to server

- gets update from server



Device B has some update for you…

I have some update..

UPDATED JSON

phiekurn8r6iyyv65t31h7ujyt73q2

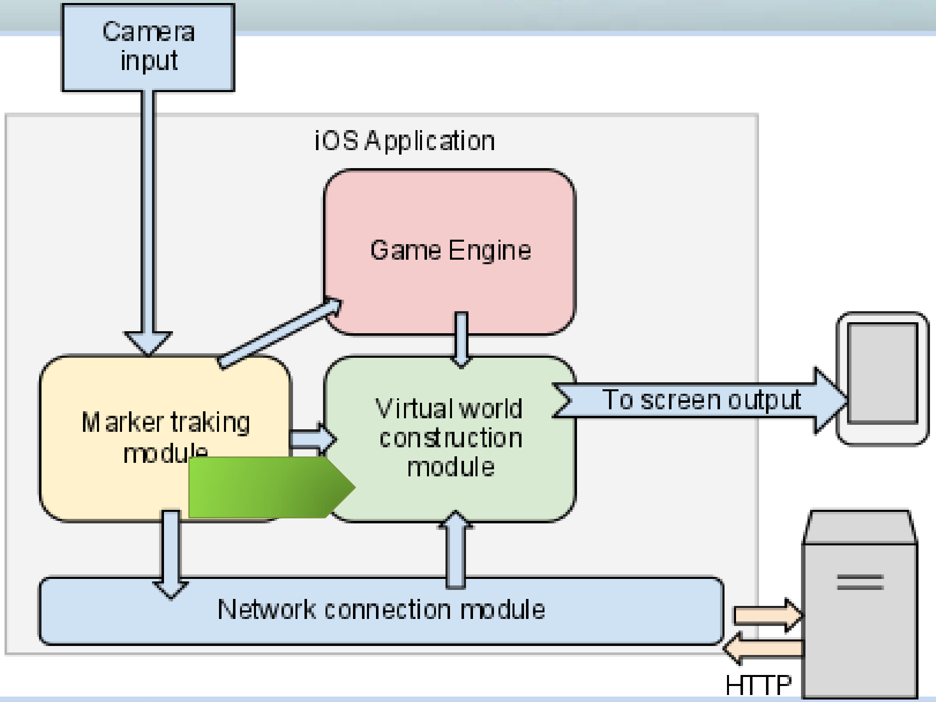
Make changes …

phiekurn8r6iyyv65t31h7ujyt73q2

# Recap

## Design

**4 main components**



# Design

## Virtual world construction

### Graphics and UI

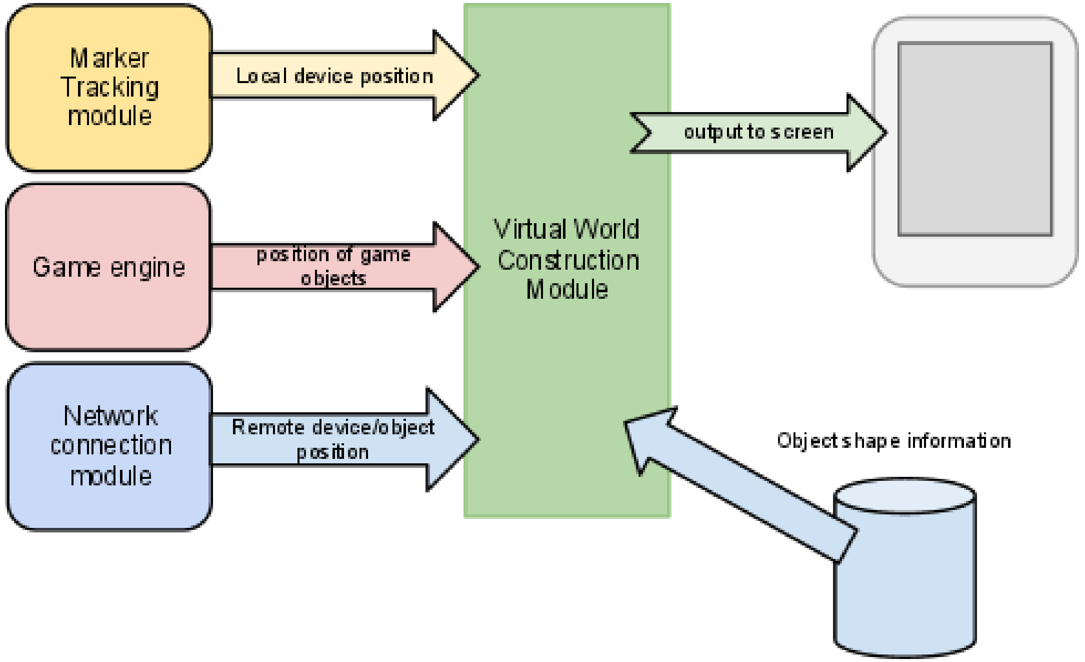
An important part to combine computer graphics and real scenes

***openGL ES (OpenGL for Embedded Systems)***

Light-weighted version of OPENGL on mobile devices

# Design

## Virtual world construction



# Design

i.Digi.T.able

## Game engine

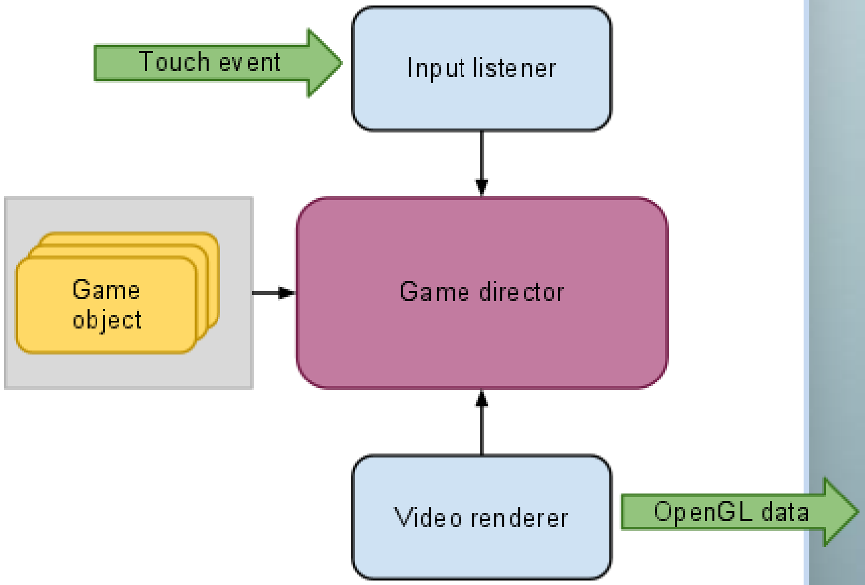
What and how you can play…

-game logic part

- Rule defined

- dependent specified game

**Overview**



# Design

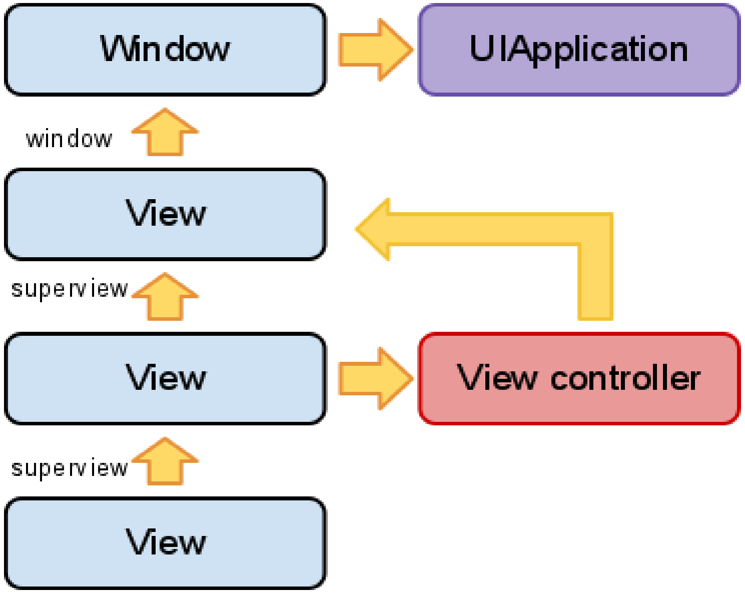
## Game engine

### Event

- first delivery to the base view

- then its inherited views

**Example of responder chain**



# Experiment

i.Digi.T.able

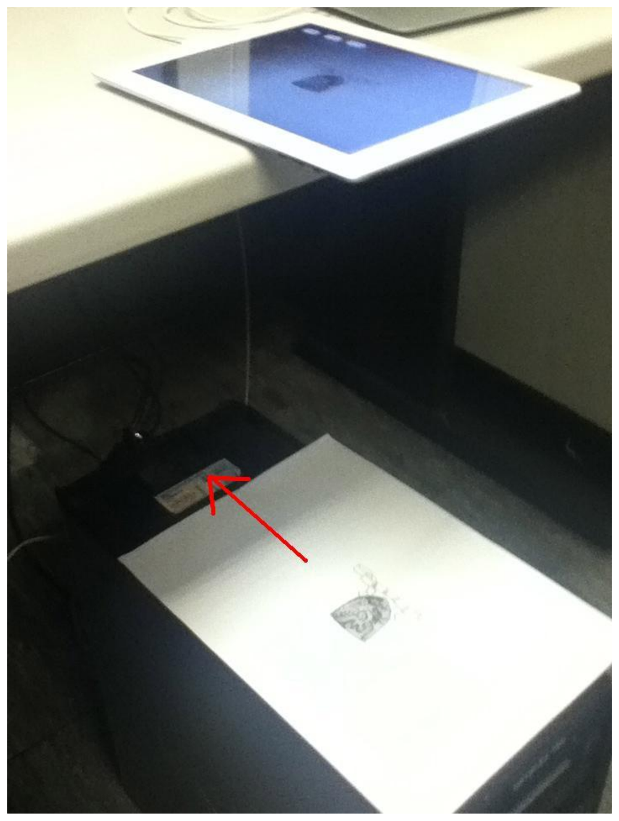
## On Camera matc

### Objective

- investigate the eff number of features on the stability of camera m

### Set up

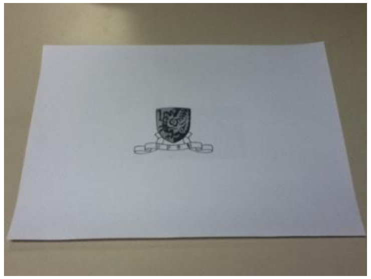
- Place a target at a f ard steadily 10cm
- Application replay ng it as a 3D model
- Observing the repl
- Test the stability of



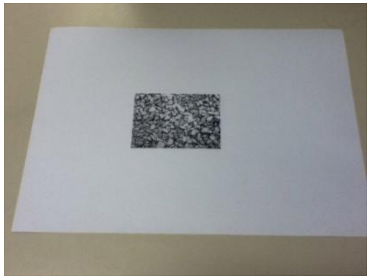
# Experiment

## On Camera match-moving

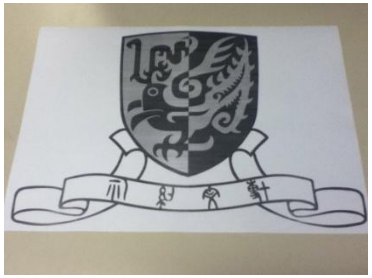
### Control



Control A
(less feature and small size)



Control B
(more feature and small size)



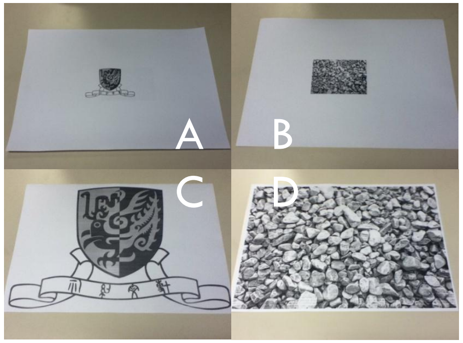
Control C
(less feature and large size)

Control D
(more feature and large size)

# Experiment

## On Camera match-moving

### Results & evaluation



- all controls, the z value keeps constant

- A and B
    - x value changes regularly , but not constantly
    - y value oscillates

-C and D
    - x value change differences are constant and small
    -Y remains constant

# Experiment

## On Camera match-moving

### Conclusion

- A large marker gives more stable tracking

- marker size also outweighs that the number of features of the marker.

**Other factors**

- Movement of the iPad

-Camera Shake

    -Algorithm to minimize unnecessary effect

# Experiment

## On Camera match-moving

### Demo program

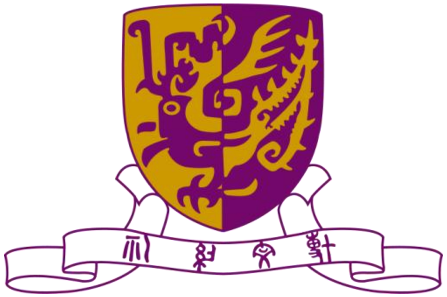
- Record the track of the movement

- Move the device

- Replay the tracked movement

DEMO

**Target marker**



# Experiment

## Network part

### Objective

-Test stability and performance

### Set up

- 2 iPads

- app installed

- 2D movement only

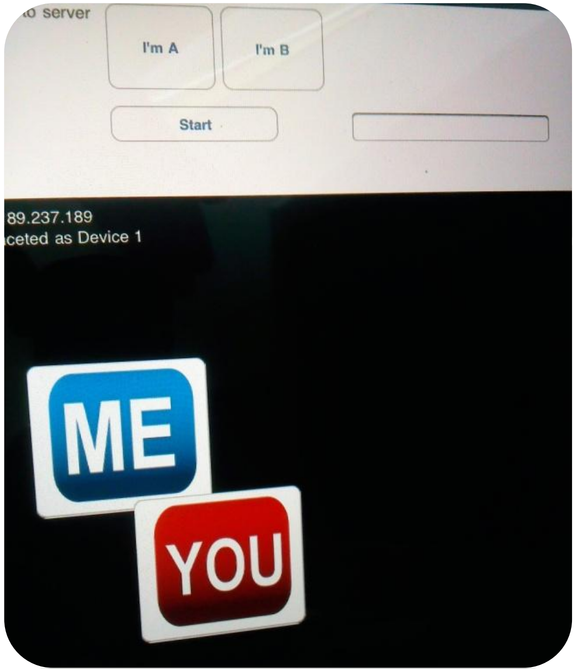
# Experiment

## Network part

### Procedure

- Connect to server
- Moves **ME** label
- Observe performance of YOU

  on another device



89.237.189
ceted as Device 1

# Experiment

Network part

DEMO

# The game

## A demo game

### Criteria

- A game to demonstrate AR and network technique
  - Marker dependent
  - battle via network
- Should be simple and direct

# The game

## A demo game

### Dodge ball

- 2-player battle version

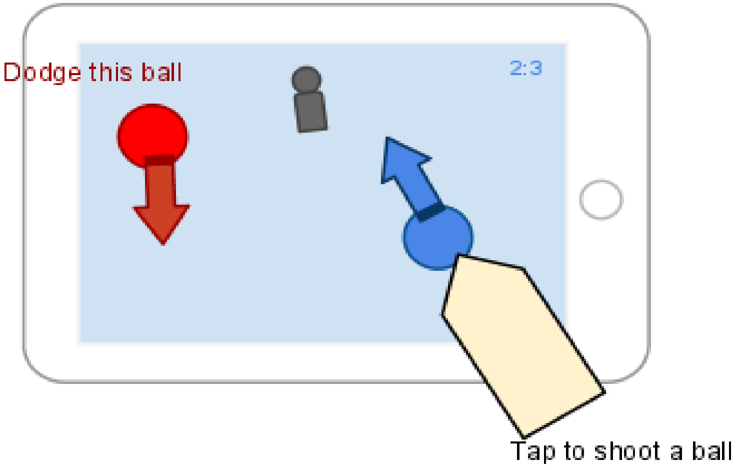
**SETUP**

-A marker on the wall

- 2 iPads with app installed

- Server ready

**Game interface**

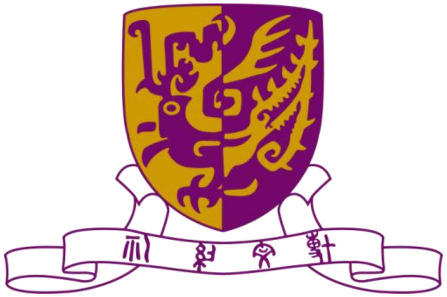


# The game

A demo game

DEMO

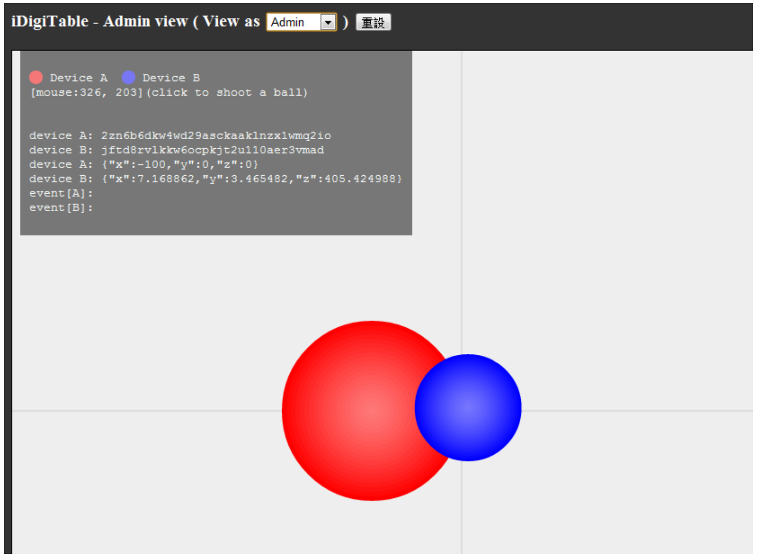
Target marker



# The game

## Admin view on web interface

DEMO



iDigiTable - Admin view ( View as Admin )

```
● Device A  ● Device B
[mouse:326, 203](click to shoot a ball)


device A: 2zn6b6dkw4wd29asckaaklnzx1wmq2io
device B: jftd8rvlkkw6ocpkjt2u110aer3vmad
device A: {"x":-100,"y":0,"z":0}
device B: {"x":7.168862,"y":3.465482,"z":405.424988}
event[A]:
event[B]:
```

# Conclusion

To summarize this semester …

-We focused on tracking AR marker and analysis positional data

- QCAR SDK

- Network implementation

- A simple dodge ball game

# Future work

## Next goals

- Stabilized camera tracking
- Network Connection
- User Interface
- Assist with iPad accessories (GPS? Gyroscope? Ambient light sensor? )
- More on Game design and implementation (snooker? chat room? )
- Investigate possibility for more clients

# Q&A Section

Q & A

# The end

i.Digi.T.able

Thank you!