COMP90018
Mobile Games

Anthony Quattrone

Developing Games

Low-level UI

- **Tasks of a low-level API**
  - Precise control about what is drawn
  - Control about the location of an item
  - Handle basic events such as key presses (see Game API)
  - Access specific keys

User Interfaces versus Games

UI is event-driven

UI is updated in response to user input

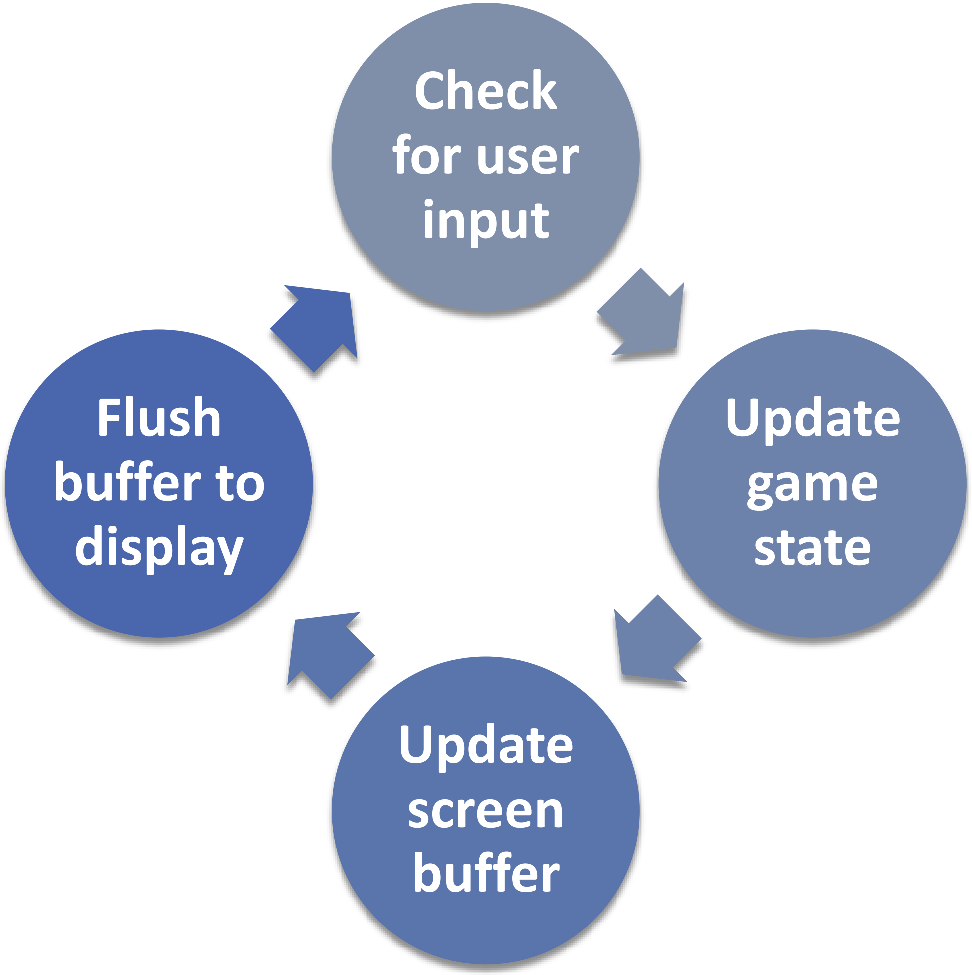
Events: pressing a soft key, selecting an item, …

Game is time-driven

Run continuously

Updates occur with and without user input

Game Loop: Main Thread



Check for user input

Update game state

Update screen buffer

Flush buffer to display

The Purpose of a Game API

- **Overview**
  - Screen buffer
  - Key polling
  - Layers
  - Sprites
  - Tiles
  - Collision detection

Screen Buffer & Layers

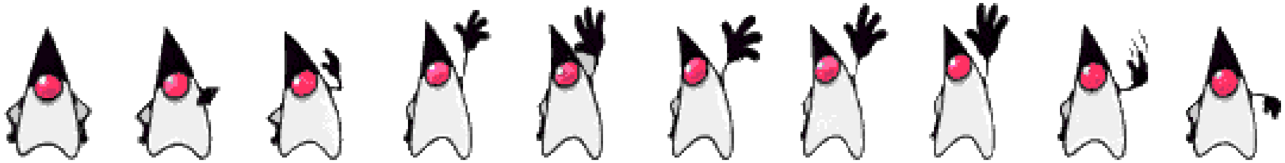
- **GameCanvas**
  - Dedicated screen buffer (*Graphics* object)
  - Supports incremental updates (instead of rendering entire frame)
  - *Flush graphics*: display contents of the buffer
- **Layers**
  - Sprites and tiled layers
  - Can be visible or invisible

Key Polling

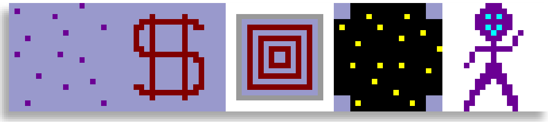
- **Query the status of keys**
  - Is a key pressed and which key is pressed
  - Duration of a key press
  - Are Keys pressed simultaneously
  - Are keys pressed repeatedly

Sprites

- **Definition**
  - Figure in 2D that is part of a larger (game) scene
  - Parts can be transparent
  - A sequence of sprites enables animation
- **Animations**
  - Frame sequence of a sprite
  - Ordered list of frames to be shown
  - Sprite is n frames: default sequence is {0, …, n-1}
  - Frames can be omitted, repeated, …



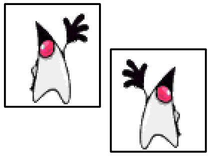
Tiles



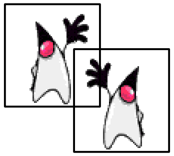
- **Why tiles?**
  - Tile is a small (rectangular) image that can be combined with other tiles to larger images
  - 2D games with large background images are composed of tiles
  - A set of tiles is small; little memory required

Collision Detection

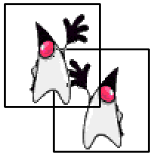
- **Collision rectangle**
  - Each sprite has a collision rectangle; usually the size of the sprite
  - Can be smaller to exclude parts of the sprite



- **Boundary-level detection (fast)**
  - Test if two collision rectangles intersect



- **Pixel-level detection (precise)**
  - Collision if opaque pixels touch



What is Different for Mobile Games

- **Processing & network**
  - Less CPU power, (usually) no hardware acceleration, less memory, intermittent network connections
- **Hardware**
  - Input capabilities, screen size,
- **Portability**
  - Sensors: location, acceleration, camera, …
  - Context-awareness, use environment as part of the game
  - Device as controller
  - Mixed reality games, location-based games, …

Tips for Good UIs

Prefer	Avoid
Relative positioning	Absolute positioning
Text extensively	Many pictures
Compress images	Large images
Reduce image size	Animations (except games)
Separate page sets	Horizontal scrolling

Usability Guidelines for (Mobile) Games

Source: http://forum.nokia.com/main/technical_services/usability/games_usability_guidelines.html

Game Start

- **Opening screen**
  - Splash screen
  - Limit the number of screens before the game start (do not annoy users)
- **Main menu**
  - Game's main menu: custom graphics
  - Avoid using UI components with standard graphics
  - Help item

Game Controls

- **General design**
  - Avoid the need for pressing two keys simultaneously: difficult on a small keyboard
  - Now: gestures …
  - One key = one command
- **In-game design**
  - Pause the game and show the main menu

Pause & Save

- **Single-player games**
  - Provide save game capability (players might have little time to play a game on a mobile phone)
  - Provide pause game capability (easily interruptible)
- **Two-player games**
  - Pause mode applies to both players
  - Provide information about why the game is paused
- **Multiplayer games**
  - Interruption of one player does not impact other players
  - Switch player to background or drop player from the game

Feedback, Feedback!

- **Status information**
  - Health, points, level, score, …
  - Not too much technical information (avoid FPS, ping, …)
- **Clear feedback on game goals**
  - Completed level, bonus level is reached, …
  - Essential elements require visual feedback: game is playable without sounds
- **Multiplayer games**
  - Who has won, who has lost
  - Show a user’s performance by using *you* instead of a name
  - Challenges: feedback that a challenge has been sent successfully

Game Experience

- **Easy to learn (!) but difficult to master**
- **Rewards**
  - Early!
  - Levels, abilities, more lives, …
  - Provide rewards randomly (motivation!)
- **Difficulty level**
  - Different settings, if possible
  - More difficult tasks
  - Do not alter game physics too much; instead more difficult tasks
  - No unbeatable AI!

Noise Pollution

- **Sound volume**
  - Default volume: close to the phone's regular sound volume
  - Enable different sound levels for background music and game sounds
  - Ability to turn sounds off quickly
  - No high-pitched sounds
- **Bluetooth multiplayer games**
  - Synchronize the background music

Distinctive Graphics

- **Avoid**
  - Small text on the screen
- **Appearance of game objects and characters**
  - Easily understood
  - Different items should look different
- **Multiplayer games**
  - We need to identify who is who (different colors)
  - But: always the same color for the same player

Post Game …

- **High score lists**
  - Provide preset results (getting into the list should not be too easy, i.e., is a reward)
  - Remember last entered name (do not force a name)
  - Server-based high-score list enables performance comparison among players
- **Easy restart**
  - E.g., *Game Over* screen: *Play again* or *Restart* command
  - Retain the previous game settings
  - Multiplayer games: quick start for a new game with same opponents

Criteria for Mobile Games

- **Easy to learn**
- **Interruptible**
- **Subscription**
  - Generate sustained revenue
- **Social interactions**
  - Massively multiplayer game, location-based services
- **Take advantage of smartphones**
  - GPS, digital camera, SMS, MMS

Optimizing Mobile Games

- **First complete the game, optimize later**
- **90/10 rule**
  - 90 percent of execution time
  - 10 percent of the code
  - Use a profiler
- **But**
  - Aim to improve the actual algorithms before resorting to low-level techniques

Why Not To Optimize

- Introduction of bugs
- Decrease of the portability of code
- Spending a lot of time for little results
- Only optimize code if the game is unplayable otherwise

Optimization Tricks I

- **Use *StringBuffer* instead of *String***
  - Any modification to a *String* variable creates a new object
- **Access class variables directly**
  - Faster than get/set methods
- **Use local variables**
  - More efficient then instance/class variables
- **Variables are more efficient than arrays**

Optimization Tricks II

- **Count down in loops**
  - Faster than counting up
- **Use compound operators**
  - Fewer byte code
- **Remove constant calculations in loops**
- **Reuse objects**
- **Assign null to unused objects & unused threads**

Mobile Pervasive Games

Pervasive Games I

- **Aim**
  - Extend the gaming experience into the real world
  - Real world: living room, public places, wilderness
- **Sensor-enabled games**
  - Accelerometer, light sensor, position, …
- **Location-based games**
  - Outdoor and indoor locating techniques
- **Augmented reality games**
  - Head mounted displays (HMDs), goggles, gloves, actuators

Pervasive Games II

- **Characteristics**
  - (Location-based) Games that are available everywhere at any time
- **Technologies**
  - Mobile devices
  - Wireless communication (3G, WiFi, Bluetooth)
  - Sensing technologies to determine player’s context, in particular identity and location

Pervasive Games: Overview

- **Benford et al.'s classification**
  - Mapping classic computer games to a real-world scenario
  - Social interaction
  - Touring artistic games
  - Educational games
- **But there is more**
  - New input devices
  - Seams

Location Sensing

- **Technologies**
  - GPS, wireless network, ultrasonic systems
  - RFID tags, accelerometers, pressure indicators
  - Vision techniques
- **Accuracy (PlaceLab)**

	Wi-Fi 802.11		GSM	
	accuracy	coverage	accuracy	coverage
Urban	20.5 m	100%	107.2 m	100%
Residential	13.5 m	90%	161.4 m	100%
Suburban	22.6 m	42%	216.2 m	100%

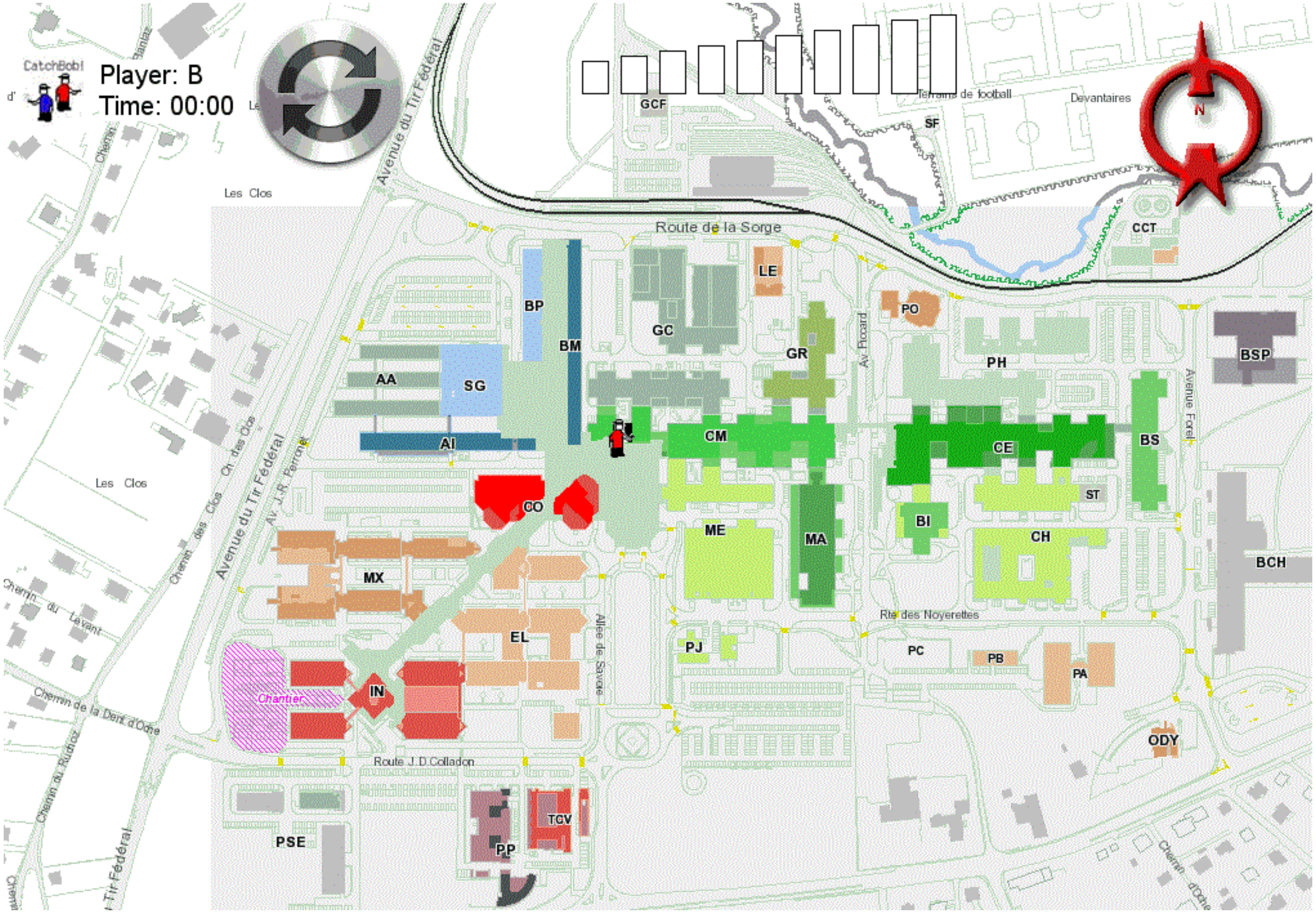
Source: http://www.placelab.org/

The Goal of Catch Bob!

- **Platform for psychological experiments**
  - How does a group benefit from knowing other members position in a collaborative task?
  - Collaborative location-based mobile game
- **Task**
  - 3 team members have to find an object and encircle it
  - Mobile device shows team members’ positions
  - Device enables communication
  - Distance to goal is given by a proximity sensor

Source: http://craftsrv1.epfl.ch/research/catchbob/

UI of Catch Bob!



Player: B
Time: 00:00

Source: http://craftsrv1.epfl.ch/research/catchbob/

Catch Bob! in Action

ECOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

CatchBob!

Nicolas Nova, Fabien Girardin, Pierre Dillenbourg
Center for Research and Support of training and its technologies (CRAFT)
Swiss Federal Institute of Technologies Lausanne (EPFL)

Images: Ecole des Arts Décoratifs de Geneve
© EPFL 2005

Source: http://craftsrv1.epfl.ch/research/catchbob/

Lessons from Catch Bob!

- **User expectations**
  - Positioning accuracy can be misunderstood
    - "I did not move physically, but I moved on the map"
    - "The proximity to Bob changed even though I did not move"
  - Intermittent network access leads users to believe that the device is faulty
  - Pre-conception about the quality of the network infrastructure and positioning systems
- **Research questions**
  - How to display bad and good positioning accuracy?
  - How to visualize network connectivity?

Source: http://craftsrv1.epfl.ch/research/catchbob/

Dealing with Imperfection

- **Uncertainty in sensing and communication**
  - Limited coverage
  - No location fix or communication available
  - Errors and jitter in measurements (sensors)
- **Approaches**
  - Remove it, i.e., choose locations carefully
  - Reveal it (but how?)
  - Exploit it, i.e., make it part of the game

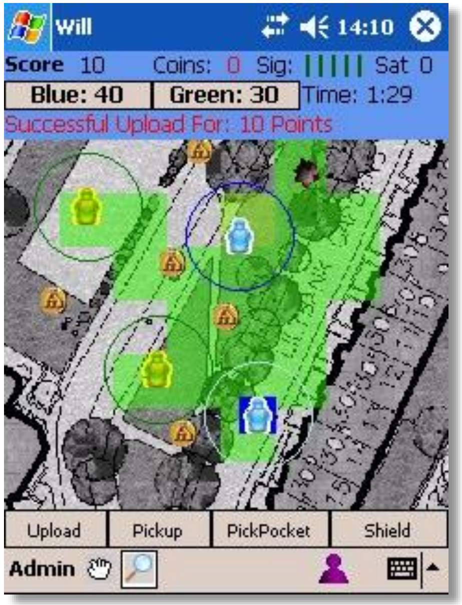
Bill: Using Seams in Mobile Games

- **Gaming on the edge: seams**
  - A seam is break, gap or ‘loss in translation’
  - Game: no wireless reception
- **Bill game**
  - Required: PDA and GPS
  - Collect coins in areas of poor network coverage
  - Upload coins to a game server in areas of good coverage: the better the coverage the higher the success
  - Player with most coins wins

Source: http://www.dcs.gla.ac.uk/~matthew/

Bill in Action

- Game rules
  - Pick pocketing: steal coins from players nearby
  - Shield: intercept pick pocketing
  - Mines: PDA is disabled for 1 minute & all coins are dropped



Will 14:10

Score 10 Coins: 0 Sig: ||||| Sat 0

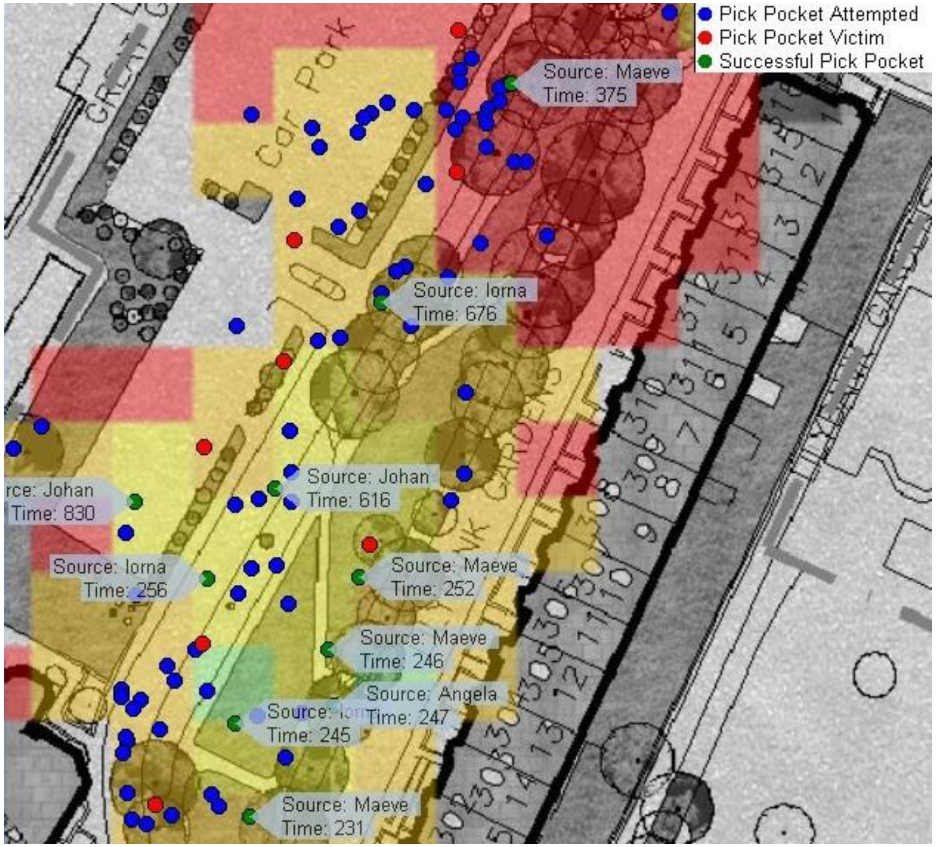
Blue: 40 Green: 30 Time: 1:29

Successful Upload For: 10 Points

Source: http://www.dcs.gla.ac.uk/~matthew/

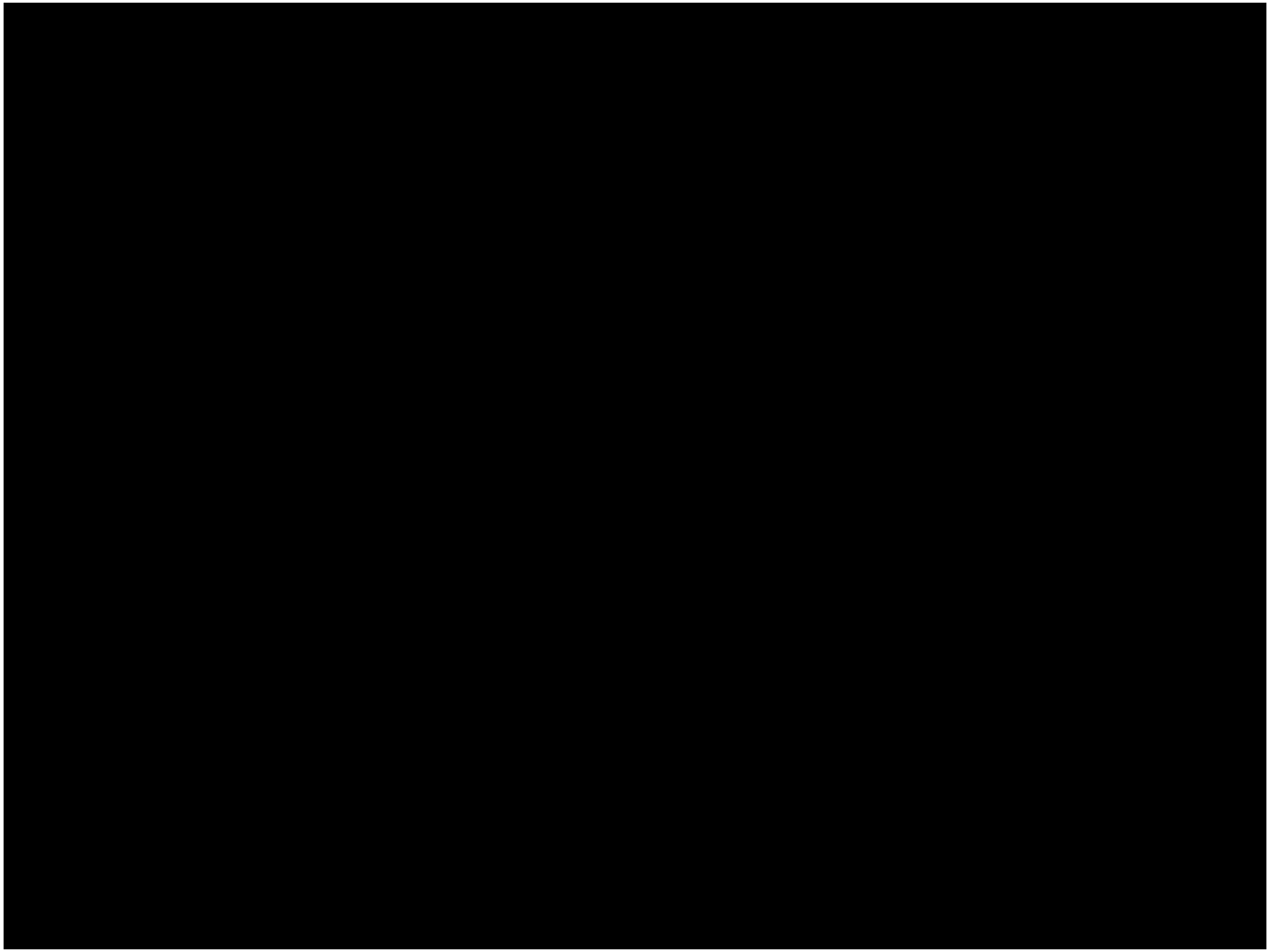
Bill: Learning Seams

- **Strategy**
  - Learn which areas are covered by the wireless network
  - Detect seams



Source: http://www.dc.gla.ac.uk/~matthew/

Bill in Action

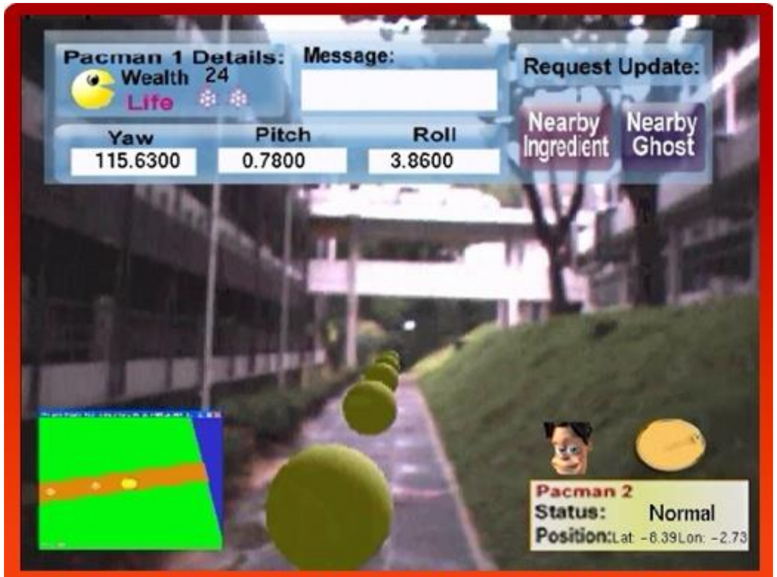


Source: http://www.dcs.gla.ac.uk/~matthew/

Human Pacman

AR Game

- Two roles: pacman and ghosts
- Players move in large outdoor environment
- Players use HMDs and wearable computers
- Overlay of virtual cookies on the real world
- Real objects have a game role



Pacman 1 Details:
Wealth 24
Life

Message:

Request Update:

Yaw 115.6300
Pitch 0.7800
Roll 3.8600

Pacman 2
Status: Normal
Position: Lat -6.39 Lon: -2.73

First-person point-of-view seen by Pacman

Source: http://www.mixedrealitylab.org/

Human Pacman in Action

Source: http://www.mixedrealitylab.org/

Research Challenges

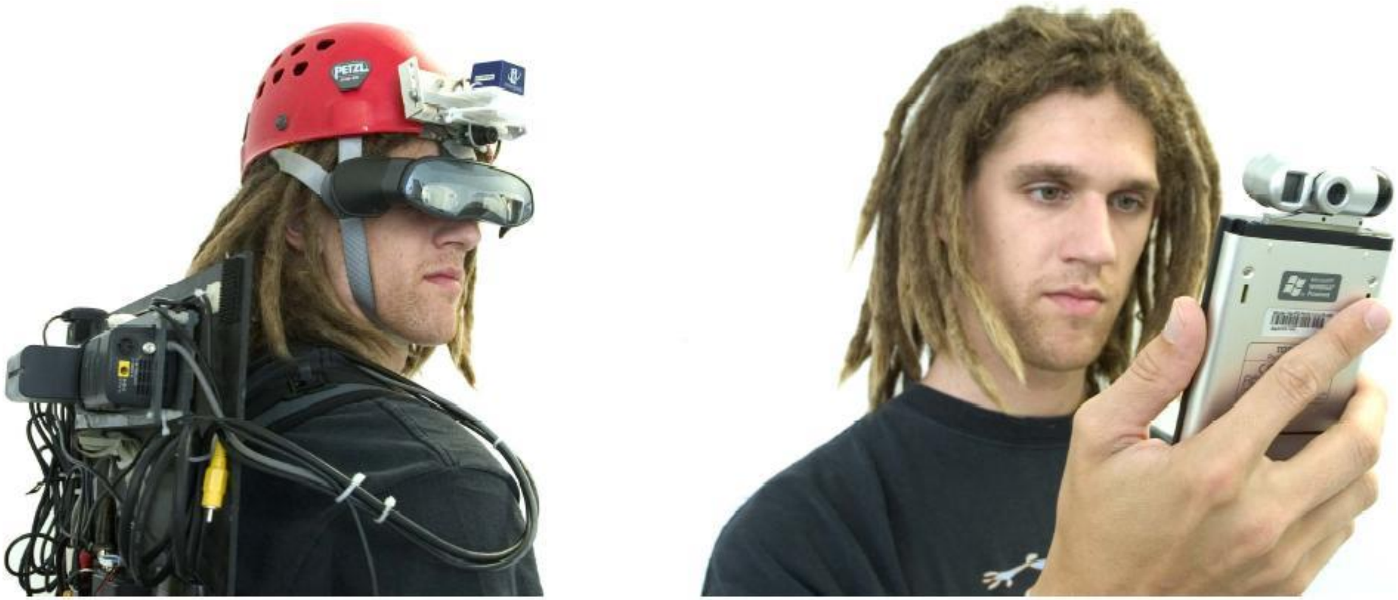
- **Hybrid architectures**
  - Client-server architectures enable a consistent game experience
  - P2P architectures enable highly localized and ad-hoc game play (pickpocketing …)
- **Integrating virtual and real domains**
  - Integration of elements of the virtual and real world
  - What should be where?

Research Challenges

- **Configuration**
  - A game has to work at different locations!
  - Seamless configuration of network connection and available sensing technologies
  - Integration of maps, images, sounds, plans, …
- **Orchestration**
  - Game provider: safety of players
  - Connection statuses, where last seen
  - How to intervene without disrupting other players?

Handheld Augmented Reality I

- **Why is this a new research area?**



Source: http://studierstube.icg.tu-graz.ac.at/invisible_train/

Handheld Augmented Reality II

- **State of the art**
  - Wearable devices are thin clients
  - Servers perform most computations (graphics rendering)
- **Multi-user AR application for handhelds**
  - Off-the-shelf PDAs
  - No infrastructure is required
  - Framework: *Studierstube* (“study”)
  - KLIMT: 3D graphics library for handhelds

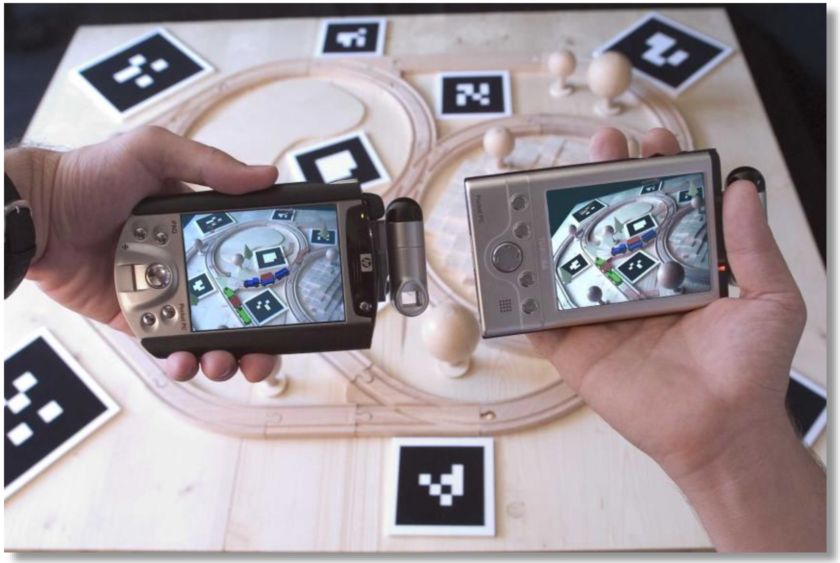
Source: http://studierstube.icg.tu-graz.ac.at/invisible_train/

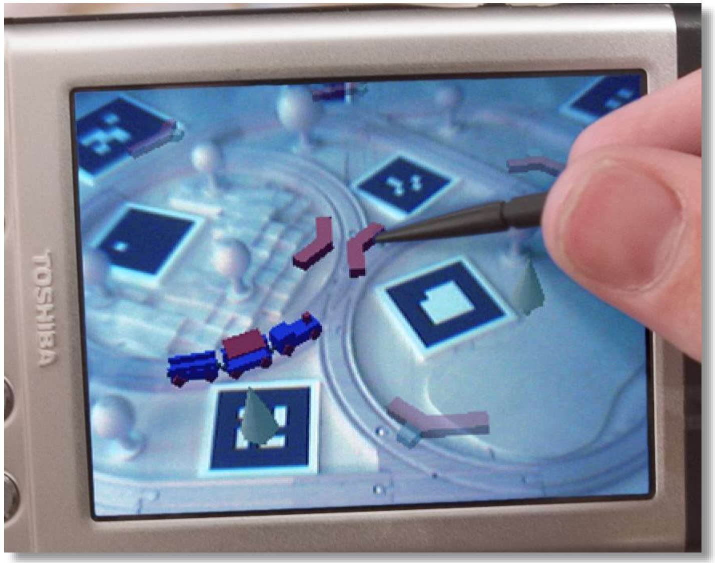
The Invisible Train

- **Collaborative multi-user AR game**
  - Players control virtual trains on a real railroad track
  - Magic lens metaphor: virtual trains are only visible to players through their PDA's video see-through display
- **Interaction**
  - Track switches & speed of the virtual trains
  - Game state is synchronized via wireless networking
- **Goal**
  - No collision of the virtual trains

Source: http://studierstube.icg.tu-graz.ac.at/invisible_train/

The Invisible Train in Action I





The Invisible Train in Action II

the invisible train

a collaborative handheld augmented reality game

daniel wagner

thomas pintaric

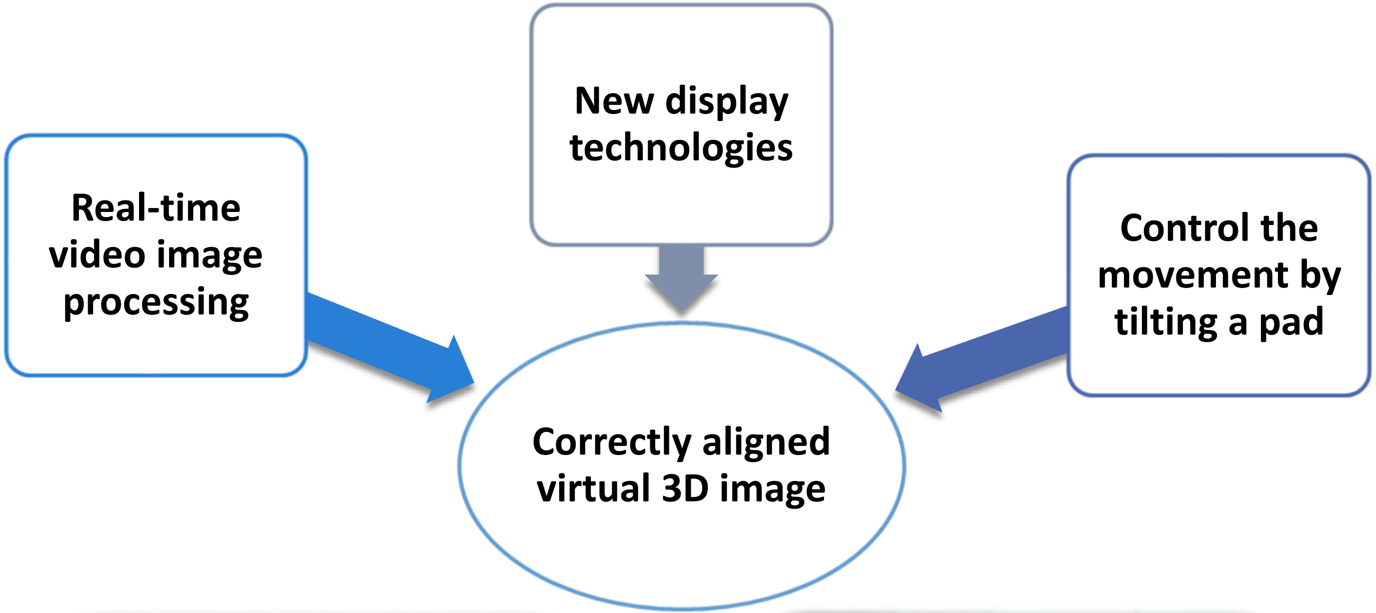
dieter schmalstieg

TUG Graz University of Technology

Erzherzog-Johann-University

Source: http://studierstube.icg.tu-graz.ac.at/invisible_train/

TiltPad Pacman



Real-time video image processing

New display technologies

Control the movement by tilting a pad

Correctly aligned virtual 3D image

Source: http://www.mixedrealitylab.org/

TiltPad Pacman in Action

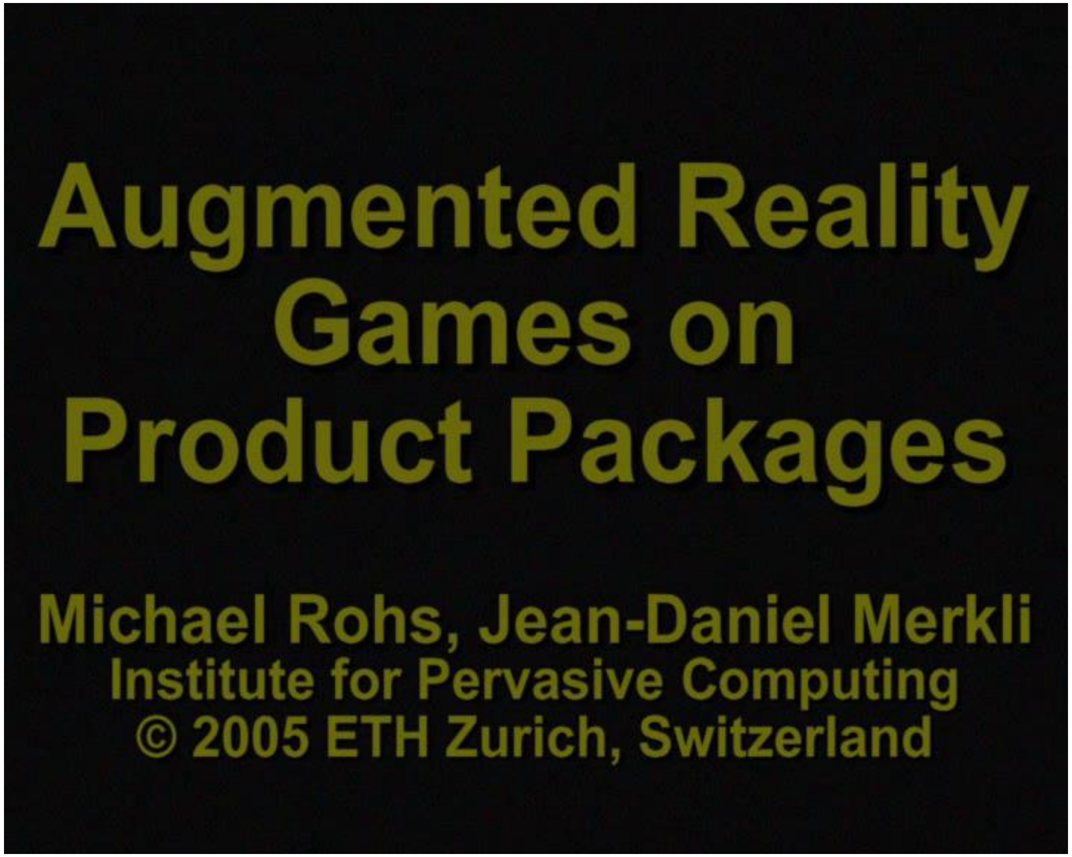
Source: http://www.mixedrealitylab.org/

Smart Product Packaging

- **Link AR games with packing**
  - Packing provides the visual background for an AR games and a visual code
  - Mobile device is used as a magic lens
- **Code**
  - Request game rules
  - Coordinate system for aligning the device

Source: http://www.vs.inf.ethz.ch/res/

Smart Product Packaging

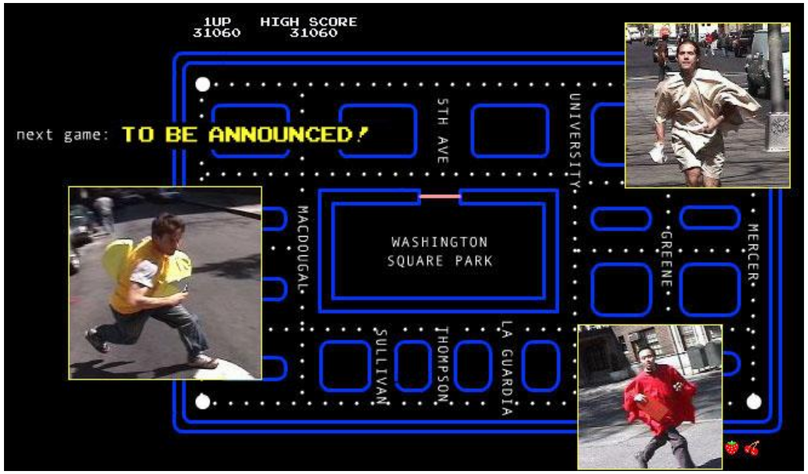


Augmented Reality
Games on
Product Packages

Michael Rohs, Jean-Daniel Merkli
Institute for Pervasive Computing
© 2005 ETH Zurich, Switzerland

Mobile Location-based Games

- **If you need more information …**
  - “Can You See Me Now?”
  - “Uncle Roy”
  - “FREQUENCY 1550”
  - Then go to www.in-duce.net/archives/locationbased_mobile_phone_games.php



Gender: Female
Hair: Long Black
Clothes:
White Other
Blue Trousers