A Software Architecture for Developing Distributed Games that Teach Coding and Algorithmic Thinking

Nearchos Paspallis, Nicos Kasenides, Andriani Piki

University of Central Lancashire (Cyprus) – UCLan Cyprus

Computer Software and Applications Conference (COMPSAC 2022)

Outline

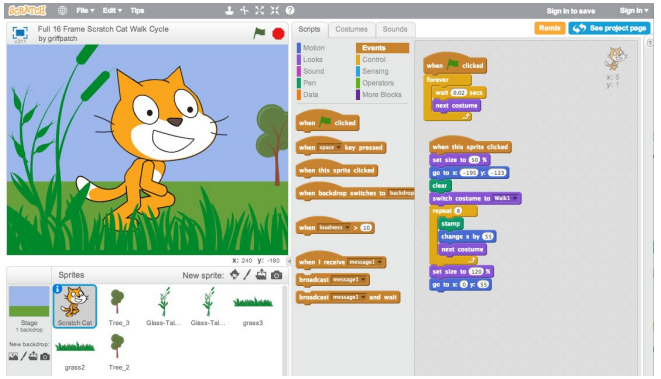
- Background and Motivation
  - Objectives and Requirements
- Software Architecture
  - Game Model and Game Play
  - Local and Distributed Architecture
- Evaluation
- Conclusions

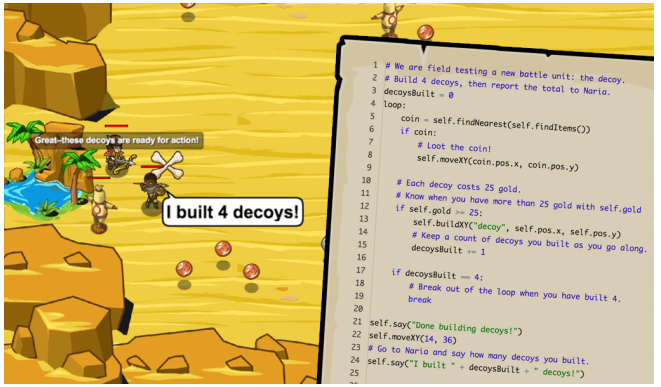
Background and motivation

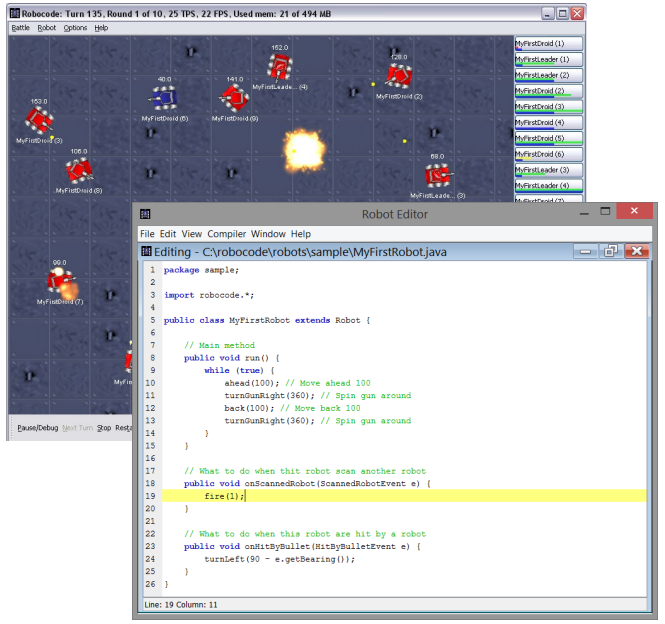
- Problem
  - Majority of people are *consumers* of the digital world
  - Very few delve into the more creative side of coding and algorithmic thinking
- Tackling the issue
  - Inspire more people to take an interest in coding and algorithmic thinking: become *producers* in the digital world
  - Hour of Code, Code Week, Code Cyprus, etc.
- Our complementary approach
  - A software architecture for developing games that teach coding and algorithmic thinking

Background and motivation

- Related works
  - Scratch platform [Resnick et al. 2009]
  - Code Combat [2021]
  - RoboCode [O'Kelly 2006]







(2021) Code combat. https://codecombat.com

J. O'Kelly and J. P. Gibson, "Robocode & problem-based learning: A non-prescriptive approach to teaching programming," SIGCSE Bull., vol. 38, no. 3, p. 217–221, Jun. 2006. https://doi.org/10.1145/1140123.1140182

M. Resnick, J. Maloney, A. Monroy-Hern´andez, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai, "Scratch: Programming for all," Commun. ACM, vol. 52, no. 11, p. 60–67, Nov. 2009. https://doi.org/10.1145/1592761.1592779

University of Central Lancashire Cyprus

Objectives

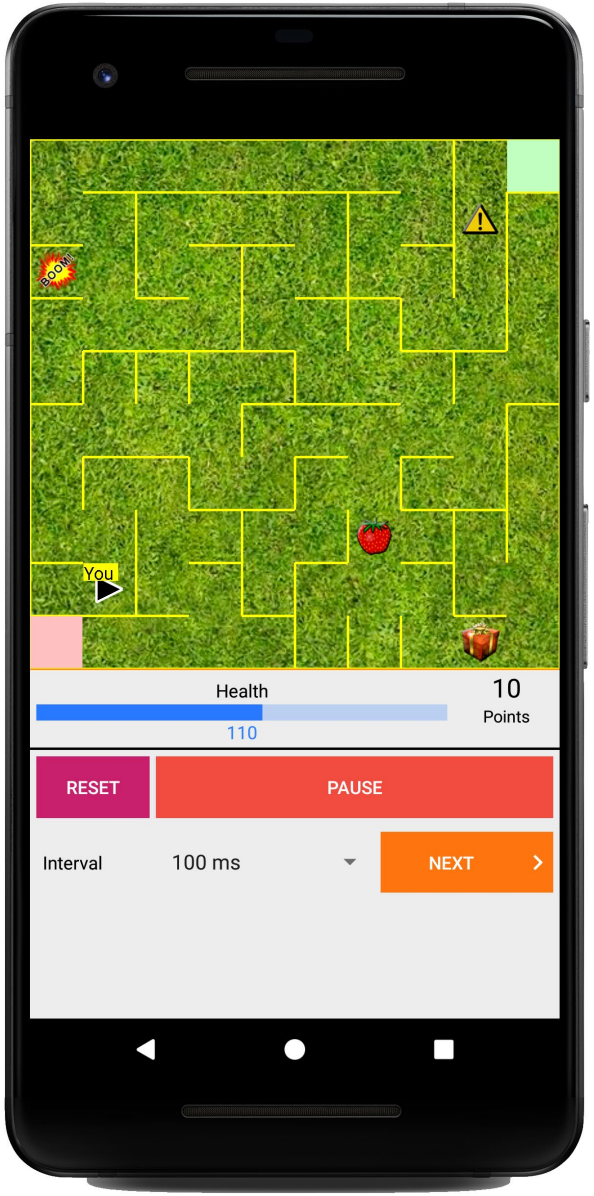
- Tackle the following research questions:
  - How can we quickly and affordably develop new games that teach coding and algorithmic thinking?
  - How can visual programming-based languages be integrated in such games?
  - How can such games be extended to enable concurrent, multiplayer mode?
  - How can we assess whether the resulting games are effective in promoting the complexity, wealth and value of learning to code?

Requirements

- Learners play simple games, which help them learn programming
- Players define code using a graphical programming language, such as Blockly
- Allow real-time competitive mode
- Run on affordable hardware like Android-based smartphones and commodity cloud platforms

Game Model

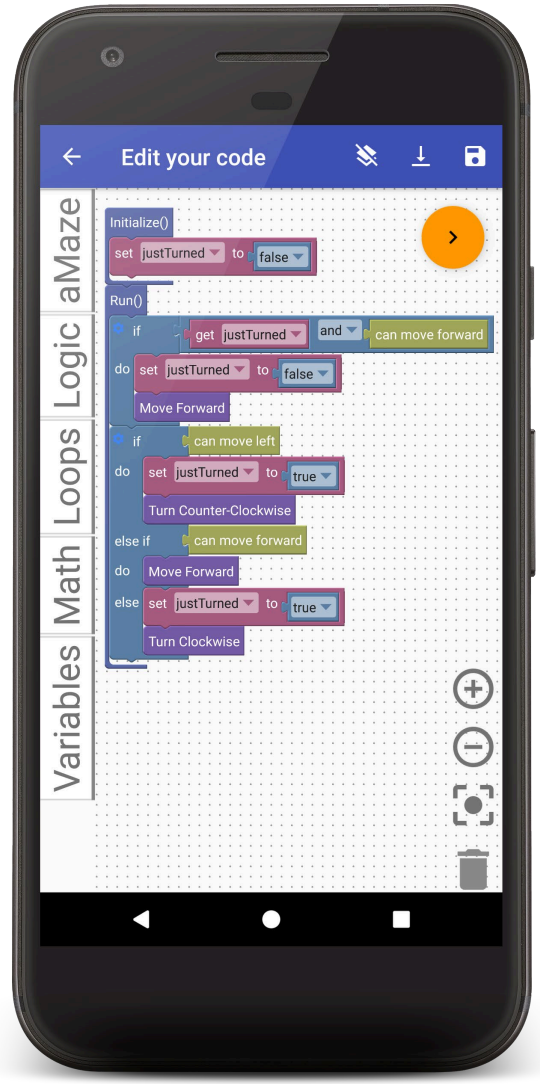
- Actors
- Grid
- Objects (pickables and obstacles)
- Game State
- Actions
- Rules



University of Central Lancashire Cyprus

Gameplay

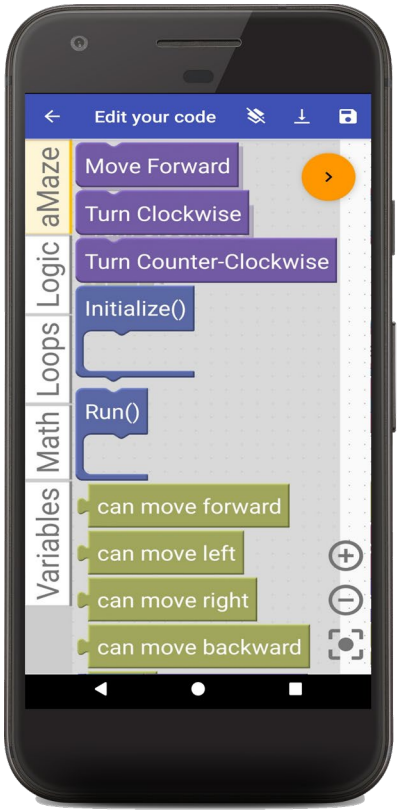
- Users play by means of defining code
- Use standard blocks from Blockly
  - **`if/else if`**
  - **`do/while`**
  - **`get/set variables`**
- Code is embedded in two methods:
  - **`Initialize()`**
  - **`Run()`**

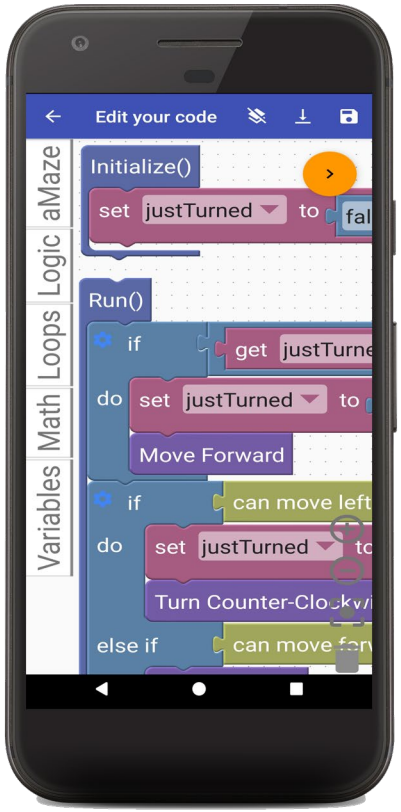


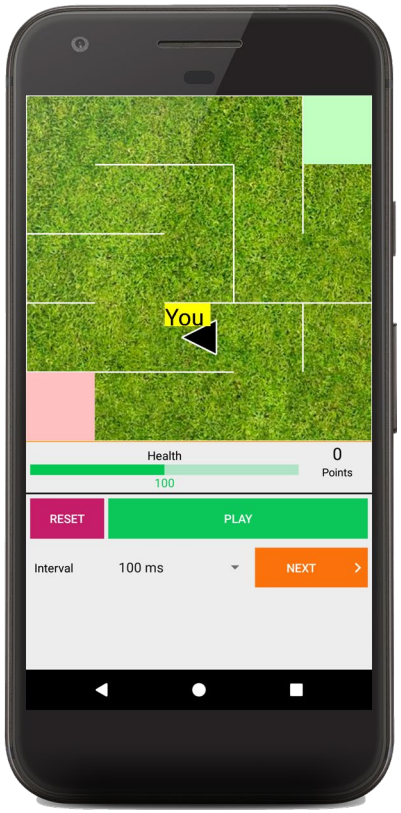
University of Central Lancashire Cyprus

Components and Architecture

- Components
  - Game API
  - Blockly Editor
  - Game Runtime







University of Central Lancashire Cyprus

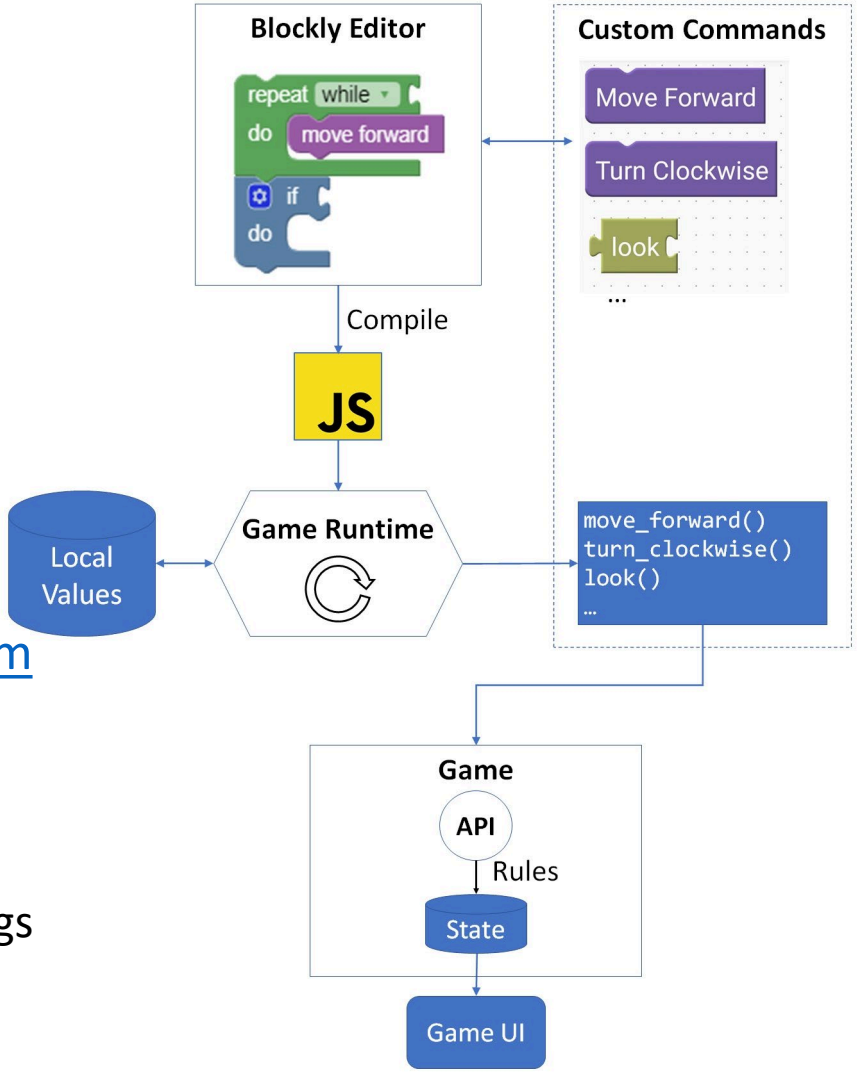
Architecture: local game

- Development steps

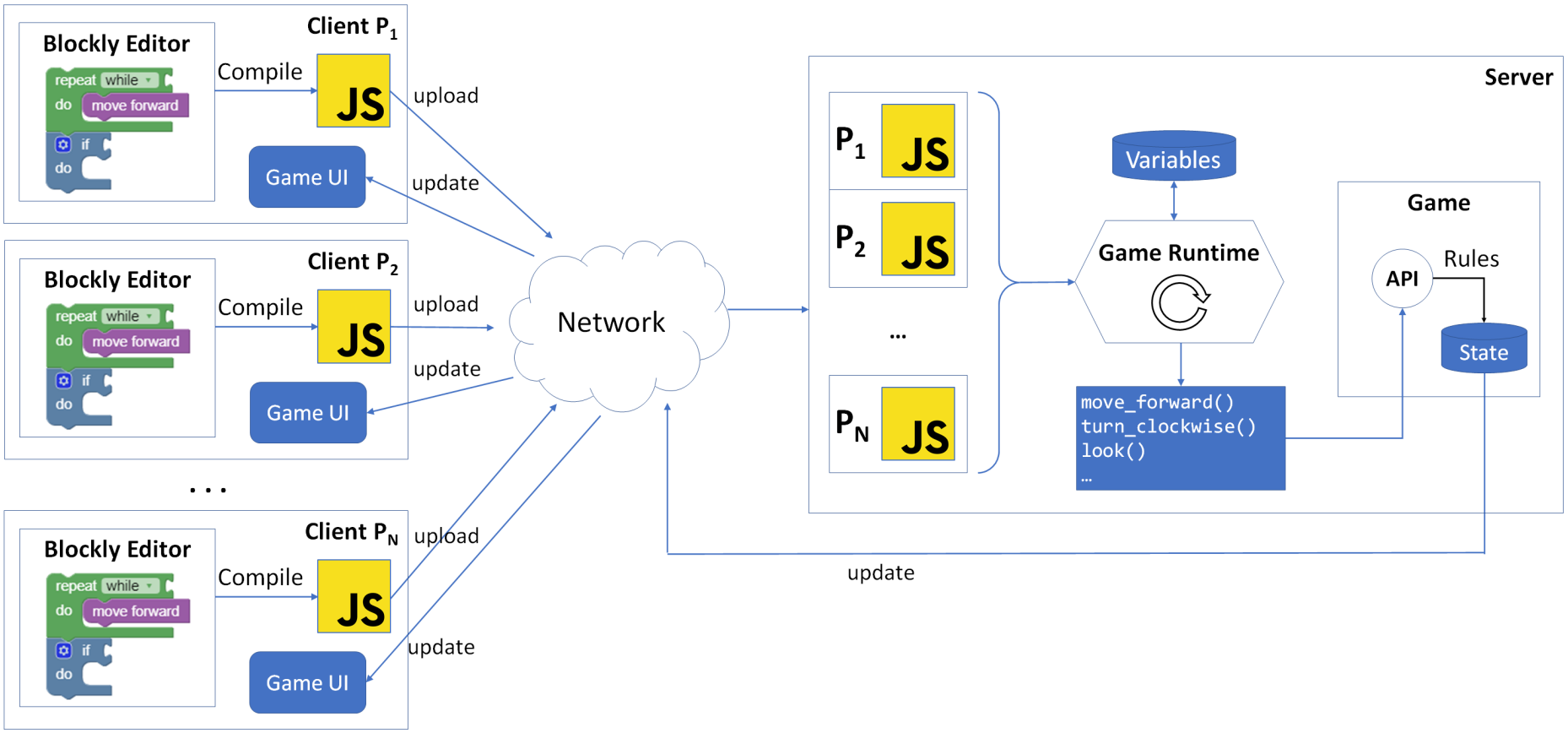
1. Define blocks for custom commands using Blockly Developer Tools (https://developers.google.com/blockly)

2. Compile into JavaScript
   - Possibly outputs errors/warnings

3. Execute in Game Runtime and send state updates to clients to update their UI

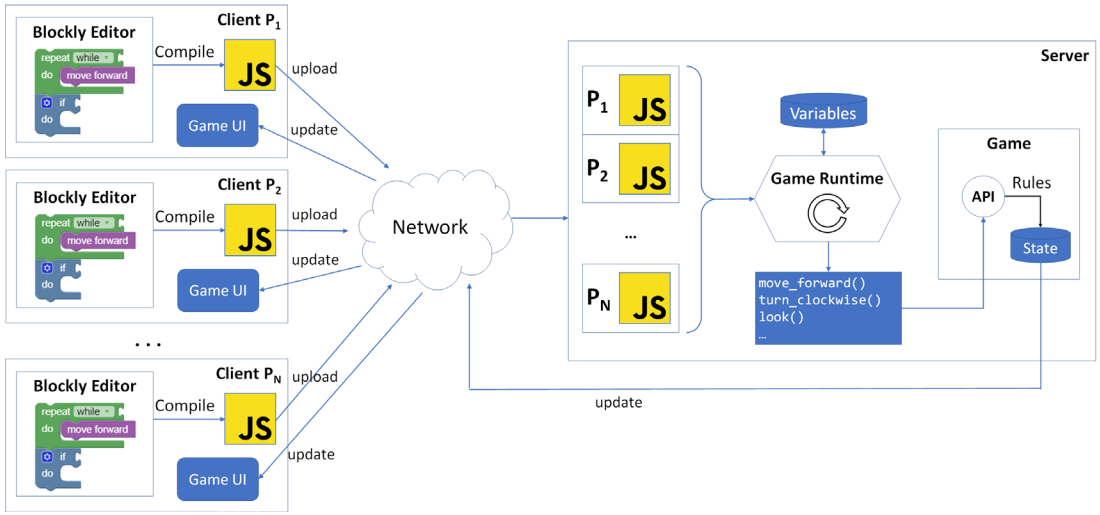


Architecture: distributed, multiplayer game



Architecture: distributed, multiplayer game

- Split between local, client-side Blockly editor and UI and a remote, server-side Game Runtime and shared state
- Development steps 1 and 2 as before (locally)



1. Define blocks for custom commands using Blockly Developer Tools
2. Compile into JavaScript
3. On the server-side, centrally execute each player's code with Game Runtime and update the UI

University of Central Lancashire Cyprus

Evaluation

- Case Study-based Evaluation
  - Reusability of code across local (single) and distributed (multiplayer) versions
  - Reusability of code across different games

Evaluation

- Case Study-based Evaluation
  - Reusability of code across local (single) and distributed (multiplayer) versions
  - Reusability of code across different games

- User-based Evaluation
  - aMazeChallenge Android game developed and used (see QRCode) Limited-scale, student-based evaluation revealed that "*the participants showed that [the app] made them feel more confident in their ability to program*", as it was documented in [Kasenides & Paspallis]

N. Kasenides and N. Paspallis, "amazechallenge: An interactive multiplayer game for learning to code," in 29th International Conference on Information Systems Development (ISD2021). Valencia, Spain: Association for Information Systems, Sep. 2021. Available: https://aisel.aisnet.org/isd2014/proceedings2021/methodologies/1

University of Central Lancashire Cyprus

Conclusions

- Achievements
  - Presented a software architecture for developing distributed games that aim to teach coding
  - Evaluation of the architecture through a case study, which demonstrates how the research questions have been answered
  - Meeting [most of] the research objectives

- Future work
  - Web-based client
  - Extend APIs, e.g., to support control of physical robots
  - Additional user-based evaluation
  - Work towards a pedagogical framework

Thank you!

Dr Nearchos Paspallis
NPaspallis@uclan.ac.uk

Dr Cand Nicos Kasenides
NKasenides@uclan.ac.uk

Dr Andriani Piki
APiki@uclan.ac.uk

University of Central Lancashire Cyprus