EdCCDroid: An Education Pilot Prototype for Introducing Code-Combat using LUA

Conor Wood[1]
Markos Mentzelopoulos[1]
Aristidis Protopsaltis[2]

[1]Faculty of Science and Technology, University of Westminster, UK
[2] Institute for Innovation in Learning Friedrich-Alexander, University Erlangen, Nuremberg

This is the published version of a paper presented at 1st Immersive Learning Research Network Conference (iLRN'15): Prague, Czech Republic, July 13-14th 2015. Published by IOS Press, Volume 19: Workshop Proceedings of the 11th International Conference on Intelligent Environments, pp.353-360, ISBN: 978-1-61499-529-6

doi:10.3233/978-1-61499-530-2-353

Workshop Proceedings of the 11th International Conference on Intelligent Environments
D. Preuveneers (Ed.)

doi:10.3233/978-1-61499-530-2-353

EdCCDroid: An Education Pilot Prototype for Introducing Code-Combat using LUA

Conor WOOD[a] and Markos MENTZELOPOULOS[a,1] and Aristidis PROTOPSALTIS[b]

[a] *Faculty of Science and Technology, University of Westminster, London, UK*
[b] *Institute for Innovation in Learning Friedrich-Alexander- University Erlangen-Nuremberg*

Abstract. The current paper present a serious game prototype developed to assist the learning of programming at a university level. The game is called EdCCDroid, and is based on Code-Combat, currently the only game field targeting audience for programming learning compared to other games that would see users touch on the purely logically side of programming without having the user entering any code. Code Combat allows users to use script languages such as javascript, Lua, python etc. as input in order to progress through a small story or compete against other players. The paper reports on a "Learn & Play" game prototype that encourages students to understand the fundamentals of programming, through algorithmic design sceptic tasks, using Robots as Avatars to perform certain tasks within the game world. The paper explores the use of the UNITY 3D libraries to design the game, the real-time interactive platform used and the instructions in Lua format. The goal of the game is to produce an attractive game theme environment as part of the game simulation concept, targeting the development of an easy use Head Up Display (HUD) for writing the equivalent task code in Lua., Feedback is provided in case of errors and a visual output of the game state is being produced with the motion/interaction of the game world-bots. The paper also reports on the usability evaluation results from a pilot study conducted with 14 participants.

Keywords. Teaching, Learning Strategies, Interactive Learning Environments, Head Up Display (H.U.D.), Unity 3D

1. Introduction

Computer games have now been around for over three decades and the term Serious Games (SG) has been attributed to the use of computer games that are thought to have educational value. Game-based learning (GBL) has been applied in a number of different fields such as medicine, languages and software engineering [1]. Furthermore SGs can be very effective as an instructional tool and assist learning by providing an alternative way of presenting instructions and content, and promote student motivation and interest in subject matter resulting in enhanced learning effectiveness [2-7]. Games are a great way for people to learn skills such as problem solving and critical thinking.

Game-based learning offers increased motivation and interest to learners through introducing fun into the learning process. Adding fun into the learning process makes learning not only more enjoyable and compelling, but more effective as well [11]. One

[1] Corresponding Author.

of the main characteristics of GBL is the fact that the instructional content is presented together with fun elements. A game that is motivating makes learners become personally involved with playing it in an emotional and cognitive way. By engaging in a dual level, learner attention and motivation is increased [12]. Systematic literature reviews [14,15] have indicated that playing computer games confers a range of perceptual, cognitive, behavioural and affective, motivational impacts and outcomes where the most frequently occurring outcomes and impacts were affective and motivational followed by knowledge acquisition/content understanding.

Research shows that that GBL and serious games has been applied in a number of different fields such as medicine, languages and software engineering Hainey, Connolly, Boyle and Stansfield [14]. Furthermore, the skills required for success in games such as thinking, planning, learning, and technical skills are skills valued as very i9mportant by employers [16]. Games are frequently cited as important mechanisms for teaching 21st century skills because they can accommodate a wide variety of learning styles within a complex decision-making context [13], foster collaboration, problem-solving, and procedural thinking [17] which are important 21st century skills. Therefore the use of serious games to teach programming seems ideal since it covers a range of skills that are essential in today's world.

Platforms like Scratch and Alice let children (and adults) create their own games and animations using simplified coding methods. Additionally a number of games have been created to teach programming to young children, such as ToonTalk [18], Kodable [19], Lightbot [9] etc.

Lightbot, developed by D. Yaroslavski in 2013 [9] is an educational single player game designed to teach people [has versions for junior (4-8) and above 9+] logically steps in programming. While the game does not involve the player entering code directly it does provide the player with a similar experience from which the player can go on to learn programming more easily. The objective within Lightbot is to use the icons provided to command the light bot to move to and light up all the blue tiles on the map. The player is given instructions towards the start of the game and a very restricted amount of commands available and number of commands to use in order to grasp the basics of the game.

Similar to Lightbot is the *Code combat* (2013) [10], an educational single and multi-player game designed to teach people to code in a multilevel-disciplinary scenarios. Code Combat allows users to input javascript, Lua, python and many more in order to progress through a small story or compete against other players.

2. Game Design for EdCCDroid

The aim of the Code Compact game is to teach the fundamental elements of programming to university students. The game was created in Unity 3D, as a PC Game, and it explores the use of the Unity 3D libraries to design the game using instructions in Lua format. The goal of player is to produce an attractive game theme environment as part of the game simulation concept, targeting the development of an easy use Head Up Display (HUD) for writing the equivalent task code in Lua., Feedback is provided in case of errors and a visual output of the game state is being produced with the motion/interaction of the game world-bots.

Within the game, Lua serves as the main interaction between the player and the Droid. Through Lua the player can direct both the movement and actions of all characters.

Some characters are static, such as doors and others have free movement (i.e.: Droid). In order to enable the player to control the characters the players/learners use an open source library called NLua. Within NLua the players use KopiLua over KerraLua libraries[2] as it is available to free Unity users. KopiLua is a C# implementation of the Lua virtual machine and thus is ideal to use with Unity and C#.

2.1. Game Head Up Display (H.U.D.) forEdCCDroid

The game makes use of a H.U.D. in order to provide the user with information about the game and to enable him to program the droids within the game. The HUD has a basic box on the screen (Figure 1, segment E) with a text editor area integrated and four buttons for the code to compile. The user initially can use the text box to enter the command (or series of commands using loops/ If statements – depending on the level) and then has a series of options: either to direct compile the code, or to save the command into a command history list (for re-usability), load previous saved command (single line command) or load a set of previous used set of instructions (multiple line script). An integrated Timer (Figure 1- segment D) provides feedback to the user for the timer elapsed for the completion of the current level, while segment area B provides the user with the level goal information. Segment area C includes the set of pre-defined set of functions – categorised for each object available in the current level, while segment area A provides a quick access to the objects on the game.

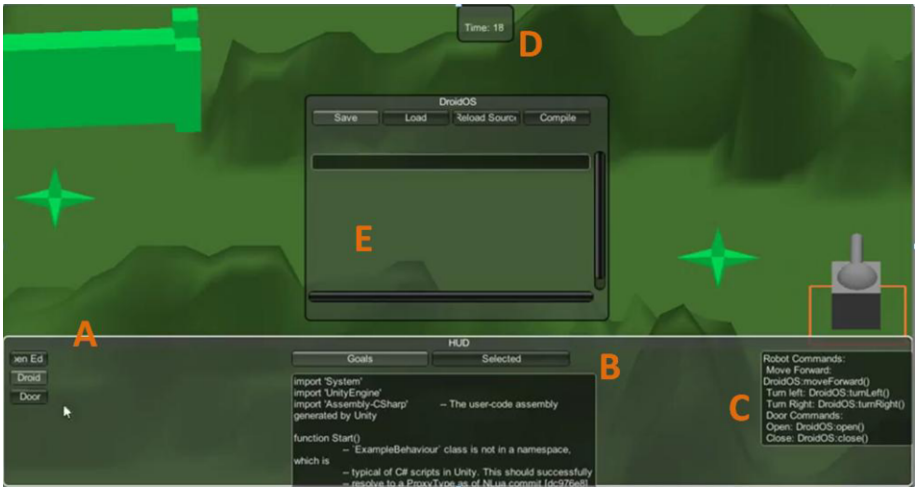

Figure 1. Main game HUD – Five main segments (A-E)

2.2. Integrating Lua programming language for EdCCDroid

To enable the player to have control over game objects, the project required scripts to be edited by the player and then run while the game was running. This required restructuring the way the project started Lua scripts. Whereas previously the scripts

[2] KopiLua is a pure C# implementation of the Lua Language and KeraLua is a wrapper to the original Lua Language written in C

were stored in the main class and run within the start function provided by Unity. On first restructuring, the compile sequence was moved into its own function and set to start running on a key press after the game had started. The project was once again restructured once a basic Heads Up Display (HUD) was created. This restructuring focused on tying the compiling code to a compile button for the user. This implementation allowed for a dynamic system of code being written during the game runtime and able to be executed.

2.3. User Controlling Object for EdCCDroid

Once the system was put into place to enable runtime code writing, the focus of implementation shifted towards allowing the player an appropriate amount of control over the game object and moving towards multiple object control. Whereas previously the player had access to the entire functionality of the game object, this functionality was stripped in favour of a controller class named "***DroidOS***".

Initially, this class was structured so that the player would reference it in their code. In turn the class would flag boolean values in the game object which would trigger movement code in the update. While this allowed for the players scripting to be more compact it would run all movements simultaneously which was not the intended behaviour for the robots. Furthermore the system would not finish running the player code until a script was compiled. For example if the player run the ***DroidOS:turnRight()*** command the game object would infinitely turn until a new script was compiled.

The solution for these issues was the use of Unity's co-routines. Calls made to the *DroidOS* would, instead of flagging boolean values, add a string to the selected characters command list, then set the character to run. The command list is a list of strings which refer to the names of the characters co-routines for actions. For example the function *DroidOS:moveForward()* adds a string to the robot of "*moveForward*" move forward is a co-routine in the robot.

In order to run a series of commands provided by the player, the characters have a managing function for all the commands in the command list. The run co-routine serves to manage the characters commands so that only one is running at any given moment, while each commands respective co-routine increases the counter which determines which command is currently running and flags a Boolean value as true until it has finished its task.

2.4. Game Commands for EdCCDroid

For the purpose of the game level scenarios, a set of Character commands were implemented for the user:

- *moveForward()* : handles both the movement and animation. Due to the nature of co-routines, the code works by incrementing the location of the robot slowly. Once the robot reaches the end of the movement the robot is no longer flagged as acting and the run co-routine will start the next command.
- *turnRight()/turnLeft()*: handles both the turning of the robot using a rotation angle based on upon a timer. The timer is incremented by the change since the last running event. This causes some minor stuttering while rotating due to the inconsistent nature of the co-routine time step, though the final result is unaffected.

- *open()/close():* Set of commands to handle the door open/close functionality. By using the lerp function over multiple iterations this provides the movement and animation for the door.
- *findTargets()/aimAtTarget : findTarget()* routines designed to locate targets on the map, and add them to the list of located objects in the robot (Level 2&3) while the *aimAtTarget()* differentiate the list of located objects into friendly/enemy targets.

3. Level Implementation for EdCCDroid Game Prototype

There are three planned levels for the current prototype, each one designed to teach a particular concept or feature of programming to the user. The purpose of level one is to teach the user the basic movement commands and introduce the user to the use of ***for loops*** to repeat movements. For this, the terrain structure for the level (maze) is intentionally lengthy to encourage the use of the loops. There are three information points on this level: The first two instruct the user about the use of for loops and how to interact with droids while the last serves to act as the end of the level (Figure 2-left). Once it has been reached the user will be taken back to the main menu where they can view their score and proceed with the game (Figure 2-right). Information point 2-Figure 2 supports an advance learning procedure to shorten the code, rather than repeat loop fragments.

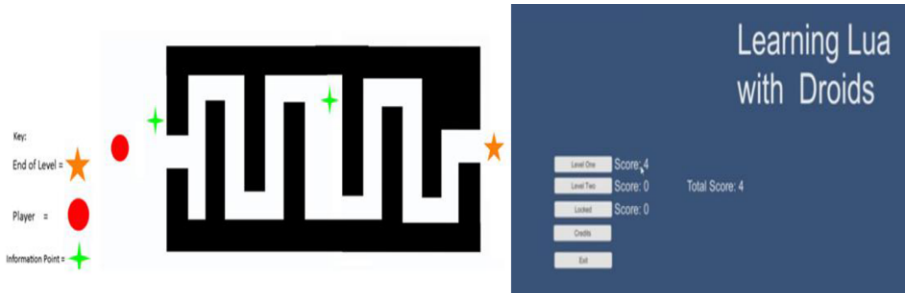

Figure 2: Level 1 EdCCDroid. Two main information points: Learning Loops (left), EdCCDroid Main Menu. Level 2 became active once player one progressed Level 1 (right)

Level two task is to teach the user about ***Arrays*** and building their own ***functions.*** For this purpose the user is provided with an open area and some targets to destroy (Figure 3). The area is however limited to stop the user from being able to exit the map. There is one information point at the start of this level that instructs the user on what to do. Once all the targets have been destroyed the user will be taken to the main menu to see their score.

Figure 3: Level 2 & 3 for EdCCDroid

Finally level three aim is to teach the user about the use of ***if statements***. The player will be presented with a similar scenario to the previous level where targets are displayed, however there will be two types of targets to differentiate, one type will be considered "*friendly*" and the other must be destroyed *("enemy").* The objective for the user is to destroy only the targets that are not coloured green. Destroying a green target will result in the user needing to restart the level. Once all the appropriate targets are destroyed the user will be taken to the main menu to see their score.

4. Evaluation procedure

The pilot evaluation of the EdCCDroid game was carried out with 14 university students (2 multimedia computing, 3 Computer science, 9 computers games development course) with experience in programming but not knowledge of the Lua programming language. Five students were 1st year undergraduates and the rest were 3rd year undergraduate students.

The study took place at the University of Westminster, London premises and each participant was tested individually. Each session lasted for approximately 20 minutes. The participants had to play the game for 15 minutes and then answer a short questionnaire. The questionnaire consisted of 18 questions in total. 17 questions was multiple choice on a Likert scale of one to five (one being the least favourable answer and the five the most favourable answer) and one open ended question. The evaluation focused on usability issues, motivational usability and usefulness of the approach.

All participants used the same apparatus: a Windows 7(64 bit) computer Intel Xeon E5506 with CPU 2.13GHz, 16GB memory and an ATI Radeon HD 5770 Graphics card.

5. Evaluation results on EdCCDroid prototype

5.1. Usability assessment

Six questions were targeted in assessing the general usability of the EdCCDroid game. The results revealed a very positive assessment regarding the usability of the game (see Figure 4). Participants found it easy to use, not very complex and they considered that they did not need to learn many things before starting to use it. Additionally they felt very confident in using the EdCCDroid game and they were willing to use it frequently in order to increase their knowledge of the Lua language.

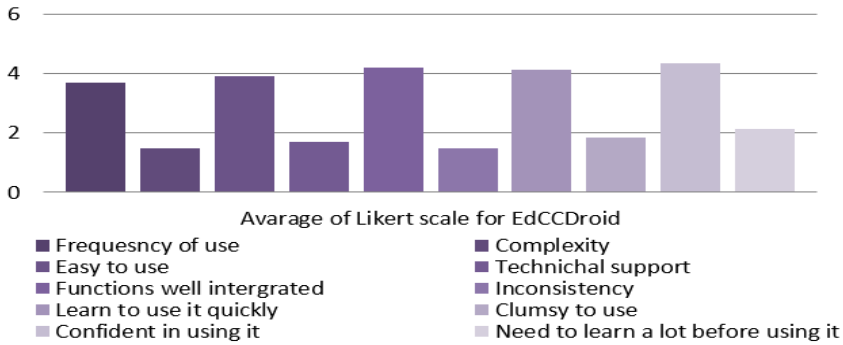

Figure 4. EdCCDroid usability results

5.2. Motivational Usability assessment

Three questions were assessing the motivational usability of the game. The results here revealed some mixed effects (see Figure 5). The participants found the experience very enjoyable (4.2 average) while at the same time they though that it does not incorporate novel characteristics. However, they argued that the game stimulates further inquiry which can be very important for learning.

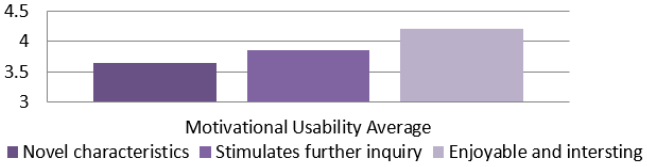

Figure 5. EdCCDroid motivational usability results

5.3. Usefulness and ease of use of EdCCDriod

The last part of the evaluation focused on the usefulness and ease of the EdCCDroid game as a teaching tool. The results here revealed (see Figure 6) that the participants found it very useful (4.6) and ideal (4.1) for teaching programming in general. They also thought again that it was not complicated to use and that it increased their team working skills.

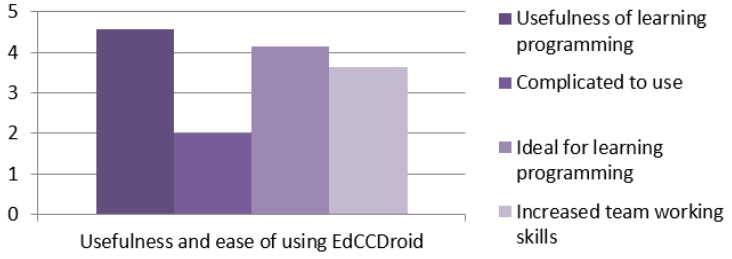

Figure 6. EdCCDroid usefulness in teaching programming

6. Discussion and Conclusions

The usability evaluation of the EdCCDroid prototype revealed some very positive results. Participants in general found it very easy to use, not complicated and they though that it was very suitable to teach programming. However, the current evaluation did not focus on the effectiveness of the learning outcome and this is a shortcoming of the study. The participants had some knowledge of programming so it is safe to assume that their experience made it easier to understand and use the game but at the same time they had not prior knowledge of the Lua scripting language and they thought that the game was a very effective approach which in fact simulated further inquiry. Thus the next steps of the work will be to assess the effectiveness of the learning outcome with the use of pre and post-test questionnaires.

References

[1] T. Hainey, T.M. Connolly, M.H. Stansfield and E.A. Boyle, "The Differences in Motivations of Online Game Players: A combined Analysis of Three Studies at Higher Education Level", *Computers and Education 2011*, **57(4)**, 2197-2211

[2] S. de Freitas and T. Neumann, "The use of 'exploratory learning' for supporting immersive learning in virtual environments", *Computers and Education 2009*, **52(2)**, 343-352

[3] S.Egenfeldt-Nielsen, "Beyond Edutainment: Exploring the Educational Potential of Computer Games", *in IT, University of Copenhagen: Copenhagen*, 1-280

[4] M. Prensky, "Don't bother me mom, I'm learning", *MN: Paragon House (2006),* St Paul

[5] K. Squire, "Replaying History: Learning World History through playing Civilization III", *in Instructional Systems Technology (2004),* Indiana University, Indiana, USA

[6] K. Squire and H. Jenkins, "Harnessing the power of games in education*", Insight (2003),* **3**, 5-33

[7] S. de Freitas, "Using games and simulations for supporting learning", *Learning, Media and Technology : Special Issue on Gaming (2006),* **31(4),** 343-358

[8] L.N. Vieira, R. Ierusalimschy, A. L. de Moura and M. Balmar, "Scriptable operating systems with Lua", *10th ACM Symposium on Dynamic Languages (2014),* 2-10

[9] D. Yaroslavski, "Lightbot", http://en.wikipedia.org/wiki/Lightbot, (2013)

[10] Code Combat, https://codecombat.com, (2013)

[11] M. Prensky, "The motivation of gameplay". *On the Horizon (2002),* ***10*** **(1)**

[12] A. Protopsaltis, L. Pannese, D. Pappa & S. Hetzner, "Serious Games and Formal and Informal Learning", eLearning Papers, elearningeuropa.info (2011), **25**

[13] K. Squire, "From content to context: Video games as designed experiences", *Educational Researcher (2006),* **35(8)**, 19-29

[14] E. Boyle, T.M. Connolly, T. Hainey, F. Hancock & J.M. Boyle, "Engagement in digital entertainment games: a systematic review", *Computers and Human Behaviour (2012)*, **28 (3)**, 771-780

[15] T.M. Connolly, E.A. Boyle, E. MacArthur, T. Hainey & J.M. Boyle, "A systematic literature review of the empirical evidence on computer games and serious games". *Computers and Education (2012)*, **59**, 661 – 686.

[16] Federation of American Scientists, "*Summit on educational games: Harnessing the power of video games for learning", (2006),* http://www.fas.org/programs/ltp/policy_and_publications/summit/Summit%20on%20Educational%20Games.pdf

[17] L. Johnson, R. Smith, H. Willis, A. Levine & K. Haywood, "The 2011 Horizon Report", *The New Media Consortium (2011)*, Austin, Texas: http://net.educause.edu/ir/library/pdf/HR2011.pdf

[18] K. Kahn, "A Computer Game To Teach Programming", *In: Spotlight on the Future, NECC '99. National Educational Computing Conference Proceedings*, (1999),

[19] https://www.kodable.com