Game Based Early Programming Education: The More You Play, the More You Learn

Ioannis Paliokas, Chistos Arapidis, and Michail Mpimpitsos

Alexander Technological Educational Institute of Thessaloniki,
Department of Information Technology, P.O BOX 141,
57400 Sindos, Greece
{ipalioka,charap,mpimpits}@teithe.gr

Abstract. Mini-languages is a branch of Educational Software for learning programming at an introductory level. On the other hand, participation, interaction and storylines make Educational Games motivating to young learners. The study presented here examined various widely known mini-languages with an emphasis on LOGO implementations and followed a combinational route to take advantage of both Game-Based Learning (GBL) and the use of mini-languages in the design of a new LOGO-like environment. PlayLOGO 3D is a video game with defined learning outcomes aiming to support GBL activities especially designed for children aged 6-13 years in the early stages of programming education. The Expert Review Method was used for initial evaluation based on a set of heuristics for usability, game play and educational effectiveness. Although the expert team found a few violations of heuristics criteria, evaluation results are very encouraging and prove that there is enough room to make programming education more fun.

Keywords: Game Based Learning, LOGO, Edugames.

1 Introduction

LOGO is widely known as a computer programming language used for programming turtle-graphics school projects. In contrast to freshmen who learn many general-purpose programming languages as tools for writing real-world application programs, LOGO is the most common educational programming language for elementary school students. It was created by Daniel G. Bobrow, Wally Feurzeig, Seymour Papert and Cynthia Solomon at 1967 for constructivist teaching [34]. LOGO has been used in the past years in education of Mathematics, Geometry, Physics and interdisciplinary approaches in all over the world. The turtle-graphics use drawing commands followed by coordinates relative to the cursor. In most applications, the cursor is depicted by a turtle or a robot. A common set of LOGO commands includes Forward, Backward, Left and Right as well as other commands to handle lists, files, functions and even recursion. Some typical tasks assigned to students are to draw basic shapes using locomotion commands on a turtle. A lot of researchers have supported the educational use of LOGO [27]. Especially about the impact the turtle metaphor has on motivating

Z. Pan et al. (Eds.): Transactions on Edutainment IX, LNCS 7544, pp. 115–131, 2013.

students, there are positive values of that flexible and universal metaphor to stimulate student's imagination, constructive, and analytical thinking [29].

On the other hand, video games constitute an alternative way to teach children of the so-called Game Generation using their own language [2]. Using a programming language to move a turtle in the screen may be fun, but this does not constitute a video game. The question is: How can we introduce programming concepts in a video game? Most educational video game designers try to emulate the commercially successful video games, but fail to gain similar success because they are resistant to change their thinking [31]. Educational software design teams usually give more importance on the visible educational and cognitive characteristics of educational applications based on their previous experience in designing educational material. This is detrimental to other characteristics that are equally important to video games like fantasy, challenge and curiosity according to the three basic game elements proposed by Malone [21]. When aimed at: A) motivating students, B) interaction with content and C) role taking, video games can increase the learning gains [11].

2 Motivation of Our Research

Why does a game-like mini-language for turtlegraphics programming is needed in schools? Mini-languages provide a sound basis for introducing programming to novices because they are small, simple, build on engaging metaphors and make user operations to be naturally visible [5]. The mini-languages approach is not entirely new. To name a few, Karel the Robot [26] was one of the first programming microworlds, Robocode [24], Gun-Tactyx [3] and Prog&Play [23] which give emphasis on Artificial Intelligence scripting. Also, Marvin's Arena [28] and MUPPETS [30] are programming games suitable for students of varying programming experience. Other implementations that could not be missing from the above list are the very well supported Alice [7] used for story-telling, animations and interactive games and C-Sheep [1] a mini-language based on a simplified ANSI C programming language.

Dealing particularly with video games, role-playing and challenge is emphasized. Interaction today is common to all contemporary educational software applications and needs no extensive analysis. Role-playing in educational activities is an established technique and when introduced in narrative interacting environments can maximize intrinsic motivation and affect positively a wide range of knowledge domains [10]. In turtle-graphics programming, role-playing can foster students to make critical choices to reach their goals. In such a scenario, students need to strategize first in order to apply knowledge to new domains as the game is going through a number of states which follow one another in a dynamic way.

Summarizing the above, most mini-languages are based on general-purpose programming languages such as Pascal, Java or C and they were designed as edutainment environments for both beginners and experienced programmers of K12 ages, while few of them were designed for university freshmen. There are rarely found competitive role-playing 3D video games based on LOGO, especially designed for elementary school students without any prerequisites or software dependencies (e.g., IDEs, compilers). The underlying motive for this implementation of LOGO was the attempt to discover new areas of interest in programming education using game-based learning scenarios.

3 LOGO Like Environments and Similar Projects

During the design phase, the development team of the PlayLOGO 3D project studied several educational software packages based on LOGO language. Studied factors include the 3D functionality, the educational orientation of the implementation, the interface design and the ease of use.

The first 'turtle' robot was created in 1969 at M.I.T., and it was based on a virtual turtle robot. It was designed to be an educational tool, primary for children. Its subject was the movement in two dimensions and this was done by typing words on a keyboard. The history of LOGO after that includes many different implementations, which belong to different categories. For example, Lego Logo is a special Logo implementation with an interesting human-computer interaction. It focuses on education, but uses Lego bricks, the well known children's toy, instead of a computer simulator. Along with the classic Lego bricks there are available special bricks that contain gears, motors and sensors used to build and program a robot. Lego Mindstorms is the successor of Lego Logo and combines everything from Lego Logo and robotics. LEGOsheets made programming with Lego Mindstorms more fun by 'continuing to reward the children with increasingly powerful abilities while requiring only small increases in the skill needed' [14].

MicroWorlds is a pure LOGO implementation for 2D turtle graphics and it became famous in Greek Elementary and Middle schools after LCSI distributed a Greek version named MicroWorlds Pro in 2002. Its basic functionality is not limited to simple movement of the turtle or creating shapes, but extends to more complicated procedural programming. Dapontes is among numerous researchers and teachers who have become enthusiasts of MicroWorlds to support programming in Greek language [8] [13].

The variety of different implementations of LOGO is impressive. A complete list of all LOGO-like environments can be found on the LOGO-Tree Project [4]. The rest of LOGO implementations examined below extends 3D functionality.

Elica appeared in 1999 by Pavel Boytchev, professor of Sofia University of Bulgaria, as one of the first LOGO implementations with 3D functionality. It can be used to visualize mathematical subjects, animate objects and create fractals. Apart from basic LOGO programming, Elica extensions allow students to experiment with design concepts and 3D animation. The build-in 3D objects library offers a starting point for 3D design and when combined with LOGO language it offers students a complete tool to build applications like a 3D chess or Towers of Hanoi.

The StarLogo is an implementation developed byMitchel Resnick, Eric Klopfer and others at the M.I.T. [6]. The most recent version is StarLogo TNG (The Next Generation), published in June 2008. It was engineered by C and Java programming languages and uses OpenGL to result in a 3D environment. The most impressive feature of StarLogo is that language elements are represented by colored blocks that fit together like puzzle pieces. Designers describe their project as a ‘programmable modeling environment for exploring the workings of decentralized systems - systems that are organized without an organizer, coor- dinated without a coordinator’ [32]. This implementation can be used to model real-life phenomena like traffic jams and market economies.

AquaMOOSE 3D, by Elliott and Bruckman [12] approached mathematics education using a desktop 3D environment and let the children play with a fish avatar that follows parametric equations in 3D. As in AquaMOOSE 3D students create mathematical challenges for one another to prevent mathophobia [12], in this project they create programming challenges to prevent programmophobia.

4 Introducing PlayLOGO 3D

In this section, theory, design philosophy and methodology of PlayLOGO 3D are described. Game play characteristics, major features and level design reveal the keypoints as they were crystalized from the prototyping processes to the final version of the game.

4.1 Game Based Learning and Constructivistic Background

The narrative metaphore is an efficient tool to maximize the learner’s motivation. The population of interest is consisting of very young learners who love story telling, use computers and play video games. Holzinger et al. say that ‘...especially small children do not make a distinction between play and learning, play and work, fantasy and reality’ [17]. In a construvistic approach games are understood as mediums to develop children’s mental and motor-sensory abilities, while the storytelling is working closely with emotions (as an expression of fantacy) to maximize the motivation to play the game.

Most game-based learning environments for programming implement a compiler or interpreter of LOGO or other programming language and have the same very specific purpose: to familiarize the user with the geometry, specifically the movement of an object, which is usually represented by a turtle. The proposed application maintains the purpose of familiarization with the movement in space and the use of LOGO commands, but it distances from the classical implementations in integrating three new parameters: A) the movement in three dimensional space, B) the existence of game mechanics and narrative and C) creating competition between users.

Among other immersive applications (educational or entertainment) which allow navigation in 3D space, this solution differs in the following key point: the movement is accurate. Using mouse or other hand-driven input devices, all moves are approximate in a sense that there is no arithmetic representation of the moving commands. In certain video games this characteristic is preferable because speed and ease are more important. Here, each move is given by typed commands and the movements are very accurate in units of length and degrees of rotation (given as parameters). In most cases, this accuracy in players movement will reveal the winner.

Players have to carefully estimate distances and to orientate in 3D space, and then carefully design a piece of LOGO code, given line by line, to reach their target. Players actively build an initial programming mental model concerning syntax, programming set, command order and visually seperate commands from parameters. This is the constructivistic core that underlyies the game mechanics and gives the game educational effectiveness.

The atmospheric scenes of the proposed game activate the curiosity and fantasy of the players and this is in line with Malone hypotheses about what makes games fun [21]. Regarding the third hypotheses of Malore, that is the challenge, this game makes the final outcome to be uncertain up to the last moment. Rules are very simple and clear to the players while the overall cognitive workload of students does not exceed a critical limit that otherwise could negatively influence the challenge [18].

The aim of educational video games is to maximize the total educational and entertainment benefits from dealing with it. In cases a video game is designed around specific educational scopes, such as the PlayLOGO 3D project, it can be harmoniously integrated in educational activities and can meet most of the goals and specifications set by the educational process. What changes need to be addressed by the traditional educational system in order to adopt the new philosophy of educational video games is outside of the scope of this paper.

4.2 Design and Prototyping

After carefully studying similar projects, the design team crystallized the basic educational and technological requirements at the initial phase of the development. In simple words, what wanted was: A) a LOGO-like environment to practice LOGO commands, B) a 3D immersive environment, C) a serious video game application.

The followed methodology was closer to Extreme Programming than traditional system development methods (such as SSADM or the Waterfall Model). Although a limited set of educational and technological requirements was determined at the beginning of the development, the small but flexible design team managed most programming and graphics design issues by avoiding lots of dependencies within the system to reduce the cost of changes. Early in the project's life cycle, brainstorming led to horizontal prototyping with a wide range of functions mostly concerning HCI and interface design for young children. For the sake of simplicity, a limited set of functions was finally chosen to be conceived analytically in a vertical prototyping procedure (Fig. 1). The most important and complicated element was the pseudo-interpreter, that is the internal machine to process the user-typed commands. The result was a number of versions, by which the most robust beta version was finally tested and distributed.

This project was engineered in the Lite-C programming language (GameStudio, v.A8, Conitec Datensysteme GmbH). 3D models were designed with SketchUp 7.0 (Google) and machinima videos were developed with iClone 3.2, 3DExchange and CrazyTalk 5 (Reallusion Inc). 2D graphics were processed with PhotoShop (Adobe Systems Inc). A data-driven development technique (DDD) was used to decouple content development from source code.

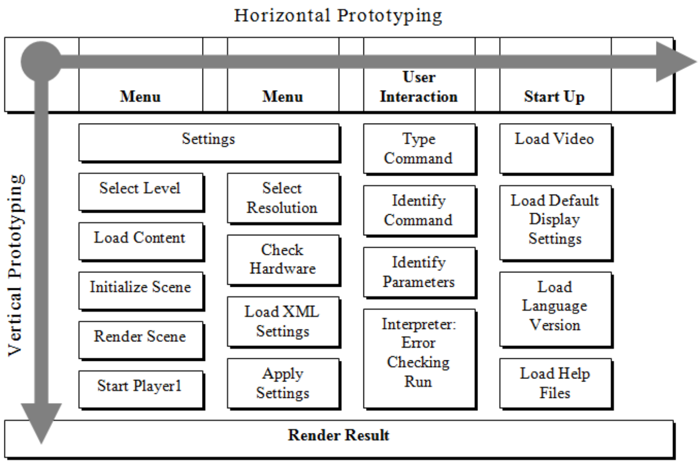

Fig. 1. Horizontal and vertical prototyping

4.3 High Concept

The scenario of the game, as described in the intro video, is a future contest for robot pilots which takes place every year in X-15 spaceship located at a constellation of Andromeda galaxy. The introductory video is used for more than one reason. Firstly, it introduces the game scenario to players (Fig. 2). This is typical to most commercial games. Secondly, the main characters (actors) explain to players the simple rules of the game in indirect way (Fig. 3). Later, players can review the help file to examine more carefully the game rules and check PlayLOGO 3D commands and syntax.

4.4 Game Overview and Features

Gameplay. Each pilot (player) drives remotely his/her robot model down in a planet's inhospitable surface (scene) while seated in an emulator at the contest platform inside the X-15 spaceship. Students play in couples and each player tries to make a collision with his/her opponent. Simple steps to reach goal are going through orientation in 3D space, lock the current position of the opponent and finally try to eliminate the distance between robots avoiding possible obstacles. Navigation is possible only by typing LOGO locomotion commands with the right syntax. During gameplay, there is no in-game vocal or textual communication between players apart from visual contact. This helps students to concentrate more on the use of LOGO locomotion commands.

The game is going through times of typing LOGO commands alternately for the two players (play in turns). After each block of commands has been typed and Enter button has been pressed, an interpretation error checking function is called. This pseudo-interpreter is also checking for data validation because some levels apply restrictions in distances and negative angles. If there are no interpretation errors, then the virtual robot executes the commands and move to a new position in 3D space. The first player who confirms a positive collision checking message from his/her robot is the winner. In this case, the other robot is destroyed and players can move to the next level. So, the collision checking of the game shows the winner depending on who sends the collision message first.

Fig. 2. Sreencapture from the intro video: The players enter the simulators

The gaming is defined as a decision making problem involving two opponent players where the outcome for each player mostly depends on the decisions taken by the other. If the current state of the game is such, one of the two players consider himself/herself as Hunter of Runaway. It is important to note that those two roles are not predefined before the game starts. Actually, it is a very sensitive and dynamically changing situation implied by the relative positions of the two players. In certain situations, one or maybe both players decide to attack because they evaluate their positions and playing order as predominant.

Avatars. Robot models are the avatars used in the game. In other words, robots are the turtles used in Microworlds and other implementations. Robots can stronger motivate the target audience and can act as a bridge between humans and machines. They are human-like in terms of body structure and at the same time they operate executing commands remotely transmitted by humans.

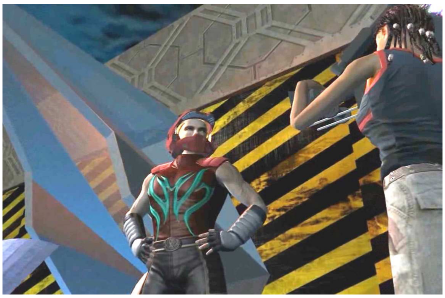

Fig. 3. Sreencapture from the intro video: The players enter the simulators

Camera. Since the environment is three dimensional and robots hold their orientation in space, the players cannot examine the whole virtual scene at any time. A mechanism independent of the robot's point of view was needed and this creates the sense of a target-free camera. By pressing the right mouse key a target-free camera is released to rotate the users point of view in all directions. This tool is used to scan the arena for the position of the opponent.

The Use of Keyboard. In 3D virtual environments like Second Life and also on commercial video games players use input devices like mouse and/or joystick to navigate. This kind of navigation is not precise because it simulates the physical movement of our bodies. In PlayLOGO 3D accuracy and quantification in navigation is a requirement because it simulates the result of a computer program, not a physical movement. This substantiates the choice of keyboard as the only input device to give locomotion commands and their parameters.

Table 1. The Complete List of PlayLOGO 3D Commands

No.	Command Name	Shortcut	Parameters
C1	FORWARD X	FD	X: distance
C2	BACK X	BK	X: distance
C3	LEFT F	LT	F: angle
C4	RIGHT F	RT	F: angle
C5	RISE X	RS	X: distance
C6	LOWER X	LO	X: distance
C7	SHIFTLEFT X	SL	X: distance
C8	SHIFTRIGHT X	SR	X: distance
C9	PASS	PS	{none}
C10	SET X	ST	X: distance
C11	FORWARDRISE X	FDRS	X: distance
C12	FORWARDLOWER X	FDLR	X: distance
C13	BACKRISE X	BKRS	X: distance
C14	BACKLOWER X	BKLR	X: distance
C15	PREVIOUS	PR	{none}
C16	CLEARSCREEN	SC	{none}
C17	PENUP	PU	{none}
C18	PENDOWN	PD	{none}
C19	PENCOLOR	PC	Color Name

Programming Set. This project is not full-featured for 3D design like other implementations (e.g. Elica). But there is the need to move in 3D space and thus new commands have to be included in the basic set of LOGO commands. Currently, there is no standardization for LOGO language by an international organization (like ISO or ECMA) as has been done in the past with other widely used programming languages. On the other hand, in most implementations, LOGO drawing (or moving) commands refer only to 2D space. As a solution, two more commands were imported from Elica: Rise and Lower. They need no more than a distance parameter to follow (integer data type). Note each turn moves the User Coordinate System (UCS) to the new position. The language structure, commands and parameters have intentionally been kept similar to Microworlds Pro, the most used LOGO environment in Greek schools. Currently the game is available in English and Greek. Not all LOGO commands have been used in the proposed project. The aim was not to replace any other official versions of LOGO language which are used in Greek Elementary and Middle school education, but to prepare students for later use of those environments to make school projects. A list of the available PlayLOGO 3D commands is showed in Table 1. Commands marked with an asterisk are available in a 'plus' version. Commands C15 to C19 are used only in design level (Raw Draw). The escape button is used to return to the Main Menu, P button for pause and right click to change camera view. Although those commands are used during the game play, it is clarified that they are used to control the game environment and they should be not considered as part of the PlayLOGO 3D programming set.

Levels. Currently, there are four levels in the game representing the corresponding arenas (Fig. 4). They are represented by futuristic scenes like surfaces of exoplanets or indoor spaceship arenas. One of them is used for training purposes before the actual contest (Raw Draw). In this extra level students can also use regular LOGO commands for drawing, plus Rise and Lower. So, the training level can be used for common LOGO drawing tasks in 3D. After a few rounds of experimentation in the training level, students get familiarize themselves with the language and syntax and can move on competition arenas.

Fig. 4. Level example: Floating Chessboard

5 Player's Experience and Expected Educational Benefits

Primarily, an educational video game needs first to be a video game. Whether it is educational, it is by educational benefits on offer and in this example, the expected ones are:

- Familiarization with the use of a programming language. Students understand that a computer language has a predefined set of commands. No other commands -not included in the set- can be used to drive a computer when this particular language is in use.

- Learn the differences between commands used to control the environment and commands which are part of the programming language.
- Understand that each command follows some rules and those rules constitute the syntax. If the syntax of a programming language is not respected, then a compilation/interpretation error will occur.
- Understand that commands can be followed by a number of parameters. Parameters can be one or more of known data types. Parameters provide the commands with data. Although some commands (like clearscreen) do not need parameters, they still can be processed by the computer to complete a task.
- Understand that a computer cannot directly execute commands typed by the user. A compiler or interpreter needs to translate the language to machine code. If the compiler/interpreter arise an error, the user gets an error message.
- Students practice on LOGO locomotion commands. This is beneficial for later use of more formal programming tools to build school geometry, math and/or programming projects.

During game time, the optimal strategy for each player is a deterministic plan of plain locomotion commands. Those commands are typed rather than given by mouse and dictate students actions in every valid state of the game. If mouse was used to move the robots as in entertainment video games, then it would be no much educational effectiveness. In this case, the language, syntax, parameters and compiling procedure would not be visible.

6 Evaluation

6.1 Usability Heuristics

Mark Griffiths [15] argues that computer games have a very positive effect on the recreational function and a remarkable success when the games are designed to address a specific problem or reason to teach a specific skill. He also remarks some negative issues that have been taken into consideration by the design team. The first refers to the fact that video games can excite and inspire students so much that finally researchers obtain false evidence as to the motives for participation and skills of participants. Moreover, the participant’s previous experience of computer games can also affect the obtained results. This makes the evaluation of this project more challenging.

Initially, it was important to formulate a set of evaluation criteria for PlayLOGO 3D. Those should be related to usability, game play and educational effectiveness. To address the above issues, a set of 40 heuristics were developed. The Expert Review Method was used to evaluate usability of the alpha version of the game prototype. User testing and expert review methods are equally accurate in case of skilful and knowledgeable usability experts [22] [19].

Nielsen and Molichs' heuristics are of the most used usability heuristics [25] for interface design. But serious games used in education have certain differences. Moreover, Korhonen et al. imply: ‘The playability heuristic set can be extended or limited based on the needs of the evaluation' [19] and here the needs extend the pleasant gaming experience. Thus, the evaluation was mostly based on the Game Playability Heuristics (GUH) of [20] -which was implemented for Mobile Games-excluding the set of heuristics related to Mobility.

Desurvire et al. [9] proposed another powerful set of Heuristics for Evaluating Playability (HEP). Based on the hypothesis that a more extensive set of heuristics does not eliminate the chances reviewers to capture criteria violations, selected heuristics proposed by HEP were used as extensions to the current set of Korhonen & Koivisto. The selection was made having in mind the game genre of the proposed application. Although both heuristics sets are complete and powerful as standalones, finally a combination was used because some heuristics were not applicable for this kind of application.

On the other hand, Korhonen & Koivisto heuristics target only on gaming characteristics. It is widely known that educational effectiveness is hard to be proved is short periods of time and especially when important educational factors are not taken into consideration, like the curricula and teachers previous experience in GBL. Nevertheless, the educational purpose of PlayLOGO 3D prototype led the design team to add another set of heuristics in order to take feedback regarding the educational effectiveness. This does not mean that no further educational evaluation is required over time. A recently proposed methodology is Playability Heuristics for Educational Game (PHEG) which is specially designed for Educational Games [16]. From PHEG, it was used only what was missing: the subset of heuristics related to Educational-Pedagogical issues. The complete(cocktail) set of heuristics used for evaluation is shown in Tables 2, and 4. The Q40 (HEP), originally located at Game Play set of heuristics was moved to Educational-Pedagogical set with a slightly different meaning. Early in case of PlayLOGO 3D means before moving to traditional LOGO-like environments for programming tasks. Let us have in mind that the proposed video game is only the first step in a wider educational pipelined procedure related to programming and does not constitute a complete educational programming environment by itself.

6.2 Evaluation Methodology

A group of four edugame experts (and teachers by themselves) played the alpha version in couples for a few rounds to discover all of the game features. They had no more than ten minutes demonstration before actual play. This short introduction time was considered enough thanks to the simplicity and the minimalistic design of the game. Later, experts were asked to take notes with clarity and cohesion. An online survey with open-ended discussion questions directly related to selected heuristics was used to collect notes. Although the questions were translated into Greek, the original English version of the questionnaire was also available to reviewers (who have at least basic written communication skills in English) to reduce the impact of possible translation errors.

6.3 Evaluation Results

All experts mentioned that graphics and the overall interface was visually appealing. Particularly, the intro video was found very helpful in order to understand differences from the more 'traditional' LOGO environments that they had previously experienced as teachers. Although answers were given as detailed notes, in a first read they were coded as positive or negative to the related heuristic. Even in cases reviewers had given controversial answers, they were asked to take position in a positive-negative manner and they did so.

Game usability results were very encouraging (Table 2). The only not 4/4 result was related to the user manual. In Q11 ('Players do not need to use a manual to play') reviewers found that reading the user manual is necessary. One reviewer mentioned that reading the manual is not a must because the game rules are very well explained in the intro video and there is an additional in-game help screen. Regarding Q2 one expert said '...the players field of view is important for pleasure and reuse. In this game there is room for improvement'. Another reviewer advises the avatars to be visually friendlier to students, assuming that the used robot models were not.

Regarding game play (Q13-Q31) experts found some violations of the used heuristics. For example in Q18 ('The first-time experience is encouraging') half of them did not found the first experience encouraging. In Q21 ('The players can express themselves') none found that players can express themselves playing that game. This result was expected, since this project was not designed to be a full featured LOGO-like environment and application development is not possible. The same is valid for Q22 ('The game supports different playing styles'), possibly because although there are different levels, the playing style is fixed.

Table 2. Usability Evaluation Results

No.	Game Usability Heuristics	ET	E1	E2	E3	E4	Viol.
Q1	Audio-visual representation supports the game	GUH	✓	✓	✓	✓	-
Q2	Screen layout is efficient and visually pleasing	GUH	✓	✓	!	!	!
Q3	Indicators are visible	GUH	✓	✓	✓	✓	-
Q4	The player understands the terminology	GUH	✓	✓	✓	✓	-
Q5	Navigation is consistent, logical, minimalist	GUH	✓	✓	✓	✓	-
Q6	Game controls are convenient and flexible	GUH	✓	✓	✓	✓	-
Q7	The game gives feedback on the player's actions	GUH	✓	✓	✓	✓	-
Q8	The player cannot make irreversible errors	GUH	✓	✓	✓	✓	-
Q9	The player does not have to memorize things unnecessarily	GUH	✓	✓	✓	✓	-
Q10	The game contains help	GUH	✓	✓	✓	✓	-
Q11	Players do not need to use a manual to play	HEP	✓	!	!	!	!
Q12	The interface should be as non-intrusive to the player as possible	HEP	✓	✓	✓	✓	-

Q: Question number, ET: Evaluation Tool, E: Expert, Viol.: Violation found

The results of Q23 (‘The game does not stagnate’) is positive because only one reviewer found a situation where a player found obstacles resulting inability for further movements. By closing the game play evaluation, one more violation found at Q31 (‘Challenges are positive game experiences, rather than a negative experience’) where experts gave controversial results. One of them said that some times experiences are positive, while some other times are not. A second one answered positively (‘so it is true to some extend’) but with doubts.

Table 3. Gameplay Evaluation Results

No.	Game Usability Heuristics	ET	E1	E2	E3	E4	Viol.
Q13	The game provides clear goals or supports playercreated goals	GUH	✓	✓	✓	✓	-
Q14	The player sees the progress in the game and can compare the results	GUH	✓	✓	✓	✓	-
Q15	The players are rewarded and rewards are meaningful	GUH	✓	✓	✓	!	!
Q16	The player is in control	GUH	✓	✓	✓	✓	-
Q17	Challenge, strategy, and pace are in balance	GUH	✓	✓	✓	✓	-
Q18	The first-time experience is encouraging	GUH	✓	✓	!	!	!
Q19	The game story supports the gameplay and is meaningful	GUH	✓	✓	✓	✓	-
Q20	There are no repetitive or boring tasks	GUH	✓	✓	✓	✓	-
Q21	The players can express themselves	GUH	!	!	!	!	!
Q22	The game supports different playing styles	GUH	✓	!	!	!	!
Q23	The game does not stagnate	HEP	✓	!	!	!	!
Q24	The game is consistent	HEP	✓	✓	✓	✓	-
Q25	The game uses orthogonal unit differentiation	GUH	✓	✓	✓	✓	-
Q26	The player does not lose any hard-won possessions	GUH	✓	✓	✓	✓	-
Q27	There is an interesting and absorbing tutorial that mimics game play	GUH	✓	✓	✓	✓	-
Q28	The game is enjoyable to replay	GUH	✓	✓	✓	✓	-
Q29	Player should not experience being penalized repetitively for the same failure	GUH	✓	✓	✓	✓	-
Q30	Easy to learn, hard to master	GUH	✓	✓	✓	✓	-
Q31	Challenges are positive game experiences, rather than a negative experience	GUH	✓	✓	!	!	!

Q: Question number, ET: Evaluation Tool, E: Expert, Viol.: Violation found

The evaluation results related to Educational-Pedagogical heuristics (Q32-Q40) where very interesting. The first question Q32 (‘Clear goal and learning objectives’) regarding clearness of objectives gave only half positive results. Two experts found that educational objectives could be clearer. One more answered positively but mentioned that there is room for improvement. The same result comes with Q39 (‘Offers the ability to select the level of difficulty’) where two experts found that arenas truly offer varying levels of difficulty. The other two found that the level of difficulty is actually the same in all arenas or there is not enough diversity as it was expected.

Table 4. Educational-Pedagogical Evaluation Results

No.	Game Usability Heuristics	ET	E1	E2	E3	E4	Viol.
Q32	Clear goal and learning objectives	GUH	✓	✓	!	!	!
Q33	The activities are interesting and engaging	GUH	✓	✓	!	!	!
Q34	Clear and understandable structure of contents	GUH	✓	✓	✓	✓	-
Q35	Can be used as self-directed learning tools	GUH	✓	✓	!	!	!
Q36	Medium for learning by doing	GUH	✓	✓	✓	✓	-
Q37	Considers the individual differences	GUH	✓	!	!	!	!
Q38	Performance should be an outcome-based	GUH	✓	✓	✓	!	!
Q39	Offers the ability to select the level of difficulty	GUH	✓	✓	!	!	!
Q40	Player is taught skills early that you expect the players to use later, or right before the new skill is needed	GUH	✓	✓	✓	!	!

Q: Question number, ET: Evaluation Tool, E: Expert, Viol.: Violation found

The last question Q40 (‘Player is taught skills early that you expect the players to use later, or right before the new skill is needed.’) gave one criteria violation. The expert found that it is possible (this was considered as a positive answer) and another explained that he was not sure.

7 Conclusions

A new solution for applying a simplified LOGO language has been presented. With PlayLOGO 3D there are neither ready solutions, nor previously stated problems. Students try to defeat one another in an interactive narrative applying LOGO commands as ‘weapons’. Its educational effectiveness is to prepare students of Elementary Education for the actual use of LOGO language in school projects and extend the LOGO philosophy beyond two dimensions. LOGO seems to be the best choice for this project because it is widely used in Public Elementary Education as a learning programming language, most teachers can use it (especially those who have not a Computer Science background) and there is a remarkable teaching experience accumulated over the past decades. As of the final visual result, the working environment has all the characteristics of a typical video game interface and the way of use is analogous to an entertainment video game.

PlayLOGO 3D is not another typical LOGO implementation to teach advanced programming issues, but a video game about LOGO. Initially, the design team was inspired by the ‘LOGO spirit’ and ‘LOGO philosophy’ that Seymour Papert described [27]. The exuberance of a commercial computer game and the characteristics of a tight turtle graphics environment were kept in balance. Star-Logo and most of other LOGO implementations, as studied earlier, offer very sophisticated environments to build applications including video games. But those LOGO implementations are not video games in their nature. They are more like Integrated Development Environments (IDEs) as members of the LOGO family because students have to learn how to apply programming principles first. In this project, students learn the very basics of LOGO without paying conscious effort and without any prerequisites, following the principles of Game Based Learning; while having fun, they empower their spatial abilities and learn what is to drive a computer using a structured language with respect to language syntax. All of the above can be said a 'programming pre-education', especially designed for students who have no previous experience in any programming language.

The currently presented PlayLOGO 3D (and future versions), the users guide and instructional materials to support students and teachers can be downloaded for free at: http://www.videotutorials.gr/playlogo3d.html. The first evaluation results are encouraging and motivate the design team for future plans. Those include the distribution of a version with more levels (arenas) and a bigger set of avatars which will be constructed by users during game time, based on a library of robot components. Currently, the Artificial Intelligence of the game is under construction in order to make possible for students to play against the computer. All future versions will keep the original characteristics of the video game without downgrading its educational scope.

Acknowledgments. Game and main menu loop music was composed by the music composer Liam Bradbury especially for this project. Scenes of intro video include objects retreived by Google 3DWarehouse and the robot models are modified models retrieved by Acknex User Magazine, vol. 68.

References

1. Anderson, E.F., McLoughlin, L.: Critters in the classroom: a 3D computer-gamelike tool for teaching programming to computer animation students. In: ACM SIGGRAPH Educators Program, pp. 7–15. ACM Press, New York (2007)
2. Baer, L.: The generation gap: bridging learners and educators. J. International Digital Media & Arts Association 2, 47–52 (2005)
3. Boselli, L.: GUN-TACTYX (January 16, 2013), `http://apocalyx.sourceforge.net/guntactyx`
4. Boytchev, P.: The Logo Tree Project, versions 1.92 (2012) (January 16, 2013), `http://www.elica.net/download/papers/LogoTreeProject.pdf`
5. Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., Miller, P.: Minilanguages: a way to learn programming principles. J. Education and Information Technologies 2, 65–83 (1997)
6. Colella, V., Klopfer, E., Resnick, M.: Adventures in modeling: exploring complex, dynamic systems with StarLogo. Teachers College Press, New York (2001)
7. Cooper, S., Dann, W., Pausch, R.: Teaching objects-first in introductory computer science. In: 34th SIGCSE Technical Symposium on Computer Science Education, Reno, NV, pp. 191–195 (2003)
8. Dapontes, N., Ioannou, S., Mastroyiannis, I., Tzimopoulos, N., Tsovolas, S., Alpas, A.: The Teacher as a Creator: Ideas on How to Teach MicroWorlds Pro in Kindergarten and Primary School Students. Kastaniotis Publications, Athens (2003)

9. Desurvire, H., Caplan, M., Toth, J.A.: Using Heuristics to Evaluate the Playability of Games. In: CHI Conference on Human Factors in Computing Systems, pp. 1509–1512. ACM Press, Vienna (2004)
10. Dickey, M.D.: Game Design and Learning: a Conjectural Analysis of How Massively Multiple Online Role-playing Games Foster Intrinsic Motivation. J. Education Tech. Research Dev. 55, 253–273 (2007)
11. Dondlinger, M.: Educational Video Game Design: a Review of the Literature. J. of Applied Educational Technology 4, 21–31 (2007)
12. Elliott, J., Bruckman, A.: Design of a 3D Interactive Math Learning Environment. In: International Conference on Designing Interactive Systems, pp. 64–74. ACM Press, London (2002)
13. Glezou, K., Grigoriadou, M.: Engaging Students of Senior High School in Simulation Development. J. Informatics in Education 9, 37–62 (2010)
14. Gindling, J., Ioannidou, A., Loh, J., Lokkebo, O., Repenning, A.: LEGOsheets: a Rule-based Programming, Simulation and Manipulation Environment for the LEGO Programmable Brick. In: Visual Languages, pp. 172–179. IEEE Computer Society Press, Darmstadt (1995)
15. Griffiths, M.: The Educational Benefits of Videogames. J. Education and Health 20, 47–51 (2002)
16. Hasiah, M., Bangi, S., Azizah, J.: Conceptual Framework for a Heuristics Based Methodology for Interface Evaluation of Educational Games. J. of Computer and Information Science 3, 211–219 (2010)
17. Holzinger, A., Pichler, A., Maurer, H.: Multi Media E-learning Software TRIANGLE Sase-study: Experimental Results and Lessons Learned. J. of Universal Science of Technology in Learning, 61–92 (2006)
18. Köffel, C., Haller, M.: Heuristics for the Evaluation of Tabletop Games. In: CHI Workshop: Evaluating User Experiences in Games, pp. 233–256. ACM Press, Florence (2008)
19. Korhonen, H., Paavilainen, J., Saarenenpaa, H.: Expert Review Method in Game Evaluations Comparison of Two Playability Heuristic Sets. In: MindTrek Conference, pp. 74–81. ACM Press, Tampere (2009)
20. Korhonen, H., Koivisto, E.: Playability Heuristics for Mobile Games. In: The 8th Conference on Human-Computer Interaction With Mobile Devices and Services, pp. 9–16. ACM Press, Helsinki (2006)
21. Malone, T.: Toward a Theory of Intrinsically Motivating Instruction. Cognitive & Science 4, 333–369 (1981)
22. Molich, R., Dumas, J.S.: Comparative Usability Evaluation. Behaviour & Information Technology 27, 263–281 (2008)
23. Muratet, M., Torguet, P., Viallet, F., Jessel, J.P.: Experimental Feedback on Prog&Play, a Serious Game for Programming Practice. Computer Graphics Forum, The Eurographics Association and Blackwell Publishing Ltd. 30, 61–73 (2011)
24. Nelson, M.: Robocode. IBM alphaWorks (2001)
25. Nielsen, J., Molich, R.: Heuristic Evaluation of User Interfaces. In: The CHI 1990 Conferenxe, pp. 249–256. ACM Press, Seattle (1990)
26. Pattis, R.E.: Karel the Robot: a Gentle Introduction to the Art of Programming. John Wiley & Sons (1981)
27. Papert, S.: What is Logo and Who Needs It? Logo Computer Systems Inc., Quebec (1999)
28. Pech, S.: Marvin's arena (January 16, 2013), `http://www.marvinsarena.com`

29. Petrovǐc, P.: Mathematics with Robotnăcka and Imagine Logo. In: The Eurologo, pp. 353–360. Warsaw (2005)
30. Phelps, A.M., Bierre, K.J., Parks, D.M.: MUPPETS: Multi-user Programming Pedagogy for Enhancing Traditional Study. In: Proceedings of CITC4 2003, Lafayette, Indiana, pp. 100–105 (2003)
31. Shelton, B.E., Wiley, D.: Instructional Designers Take All the Fun Out of Games: Rethinking Elements of Engagement for Designing Instructional Games. In: Annual Meeting of the American Educational Research Association, San Francisco (2006)
32. Introduction to StarLogo (January 16, 2013), http://education.mit.edu/starlogo/
33. Wang, D.L., Li, J., Dai, G.: Usability and Internationalization. In: Aykin, N. (ed.) HCII 2007. LNCS, vol. 4559, pp. 622–630. Springer, Heidelberg (2007)
34. Logo (January 16, 2013), http://en.wikipedia.org/wiki/Logo_ (programming_language)