Evaluation of Computer Games Developed by Primary School Children to Gauge Understanding of Programming Concepts

Amanda Wilson
Thomas Hainey
Thomas Connolly
University of the West of Scotland
amanda.wilson@uws.ac.uk
thomas.hainey@uws.ac.uk
thomas.connolly@uws.ac.uk

Abstract

Under the Curriculum for Excellence (CfE) in Scotland, newer approaches such as games-based learning and games-based construction are being adopted to motivate and engage students. Construction of computer games is seen by some to be a highly motivational and practical approach at engaging children at Primary Education (PE) level in computer programming concepts. Games-based learning (GBL) and games-based construction both suffer from a dearth of empirical evidence supporting their validity as teaching and learning approaches. To address this issue, this paper will present the findings of observational research at PE level using Scratch as a tool to develop computer games using rudimentary programming concepts. A list of criteria will be compiled for reviewing the implementation of each participant to gauge the level of programming proficiency demonstrated. The study will review 29 games from Primary 4 to Primary 7 level and will present the overall results and results for each individual year. This study will contribute to the empirical evidence in games-based construction by providing the results of observational research across different levels of PE and will provide pedagogical guidelines for assessing programming ability using a games-based construction approach.

Keywords: primary education, curriculum for excellence, programming, games-based construction, evaluation, scratch, review, pedagogy

1 Introduction

Previously information and communications technology (ICT) was one area of the curriculum that Her Majesty's Inspectorate of Education reported should be improved upon within the primary school sector in Scotland to provide children with increased opportunities to use computers (HMIE, 2009). Within the Curriculum for Excellence (CfE) teachers are being encouraged to make more use of different styles of approaches to learning, one of which is the use of ICT within learning. ICT as an approach to learning is being encouraged to develop children's digital literacy skills and some suggested means of implementing this are through the use of Glow – the Scottish schools intranet system or through games-based learning (GBL) (LTS, 2011a), which is supported by the Consolarium (LTS, 2011b), an initiative set up by Education Scotland to support teachers in exploring the use of GBL in their class. This is further enhanced within the curriculum that looks for children to be making use of GBL through designing and creating their own games. Game construction is a relatively unexplored area especially within PE and a study by Vos, van der Meijden and Denessen (2011) has shown that children were more motivated to learn when constructing their own games over playing games.

This paper examines the use of game construction tools in the primary classroom to introduce children to programming. It will then present a coding scheme developed to evaluate scratch games followed by an evaluation of games created through an introductory study on game construction with Scratch for three classes in the same primary school, with an in depth look at two of the games created and then discuss the results obtained. The paper concludes with a brief discussion on future research directions.

2 Previous work

2.1 Programming for children

Programming can be taught from an early age (Gibson, 2003; Resnick et al., 2009) and over the years there has been a number of languages developed aimed at the novice user (McNerney, 2004; Kelleher and Pausch, 2005). There have been many projects undertaken to introduce programming to children, though not all necessarily within the classroom. Some have created after school clubs to introduce children to computing (Borghi, De Ambrosis and Massara, 1991; Gibson, 2003; Kelleher, Pausch and Kiesler, 2007; Lindh and Holgersson, 2007; Malan and Leitner, 2007; Maloney *et al*., 2008). Kelleher, Pausch and Kiesler (2007) created a programming environment that would engage girls more in programming by modifying the Alice platform. This was then tested using a between-subjects study for both programming environments to identify which one they enjoyed using more. Results showed that the girls did prefer the modified programming environment, however this study only concentrated on girls. The Toontalk programming environment has a video game-like style with animations within it symbolising programming attributes; e.g. a house in Toontalk represents an object or actor in programming (Kahn, 1996) and has been used in a study with pre-school children (Morgado and Kahn, 2008).

Adventure author is another example of a programming tool designed to make games and is aimed at children aged 10-14 (Robertson and Good, 2005). The tool allows children to create their own interactive story, which other children are then able to play. This is also similar to Storytelling Alice, (Kelleher, Pausch and Kiesler, 2007), which was used as an approach to encourage girls to develop an interest in computer programming and is based on Alice (a freeware object-oriented educational programming language). The use of electronic toys is another way in which children can be taught about programming, programmable bricks (Wyeth and Purchase 2000) and Lego toys such as *Mindstorms*. Using *Lego Mindstorms* children were taught some basic computer concepts over six months, with lessons structured so that they started off as being teacher-led then eventually allowed pupils to work on their own (Barrios-Aranibar et al., 2006).

2.3 Games-based construction

Van Eck (2006) suggests from a review of the literature that there are three ways of introducing GBL into educational establishments either through the students creating their own games, use of serious games or through the use of Commercial off the Shelf (COTS) games. This is in line with Seymour Papert's vision that children should be programming the computer rather than being programmed by the computer through computer-aided learning (Papert, 1980). Kahn (2007) and Kelleher and Pausch (2005) have shown that since Logo was created in 1967 many others have tried to get children programming with various other languages. Papert's theory of constructionism regards learning as a process in which the learner actively constructs his or her knowledge by interacting with the subject matter. The constructionist perspective puts game construction in the hands of children to encourage their knowledge through developing objects (Papert, 1991). In the game construction approach, the aim of the game construction tool is to support such an activity by providing an appropriate environment. Kafai (2006) suggested that constructionists have focused their efforts on providing students with greater opportunities to construct their own games and to construct new relationships with knowledge in the process, rather than embedding 'lessons' directly in games.

Game construction has been relatively unexplored within the classroom (Baytak and Land, 2011). With applications like Scratch, game construction is becoming more accessible to children (Resnick, Kafai and Maeda, 2003). Scratch is a programme that takes its inspiration from Logo (Papert, 1980). It is a visual-based tool that uses blocks that are snapped together to create scripts. It is primarily aimed at children aged around eight, however, statistics from the Scratch site show that the average age of users is twelve (MIT, 2011) and there is a wide age range of users for this tool. It was envisaged from the outset that while this project was to introduce children to computing within deprived areas and eventually the informal educational benefits of it would be studied at a later date (Resnick, Kafai and Maeda, 2003).

An exploratory study into the use of game making within a class of 30 P6 (9-10 year old) children, to develop aspects of the successful learner strand of the Scottish Curriculum for Excellence (CfE) was

carried out by Robertson and Howells (2008). The study involved a case study where six students constructed a game using the Neverwinter Nights authoring tool. The findings showed positive results of successful learning and demonstrated that participants were able to link and apply their learning to new situations. The study showed that game construction provides the opportunity for children to achieve successful learning, however to fully explore this, the lessons need to be directed by the class teacher. Denner, Werner and Ortiz (2011) conducted a study of girls and the games they created in an after school project using Stagecast Creator. The study conducted over a period of 14 months showed that using both programming and design activities can support the learning of programming concepts when creating games.

2.4 Scratch

While much of the focus on Scratch research projects is on how much the users enjoy Scratch or what blocks they are using most in creating projects there is currently little published research on what learning takes place when making games with Scratch in a primary classroom setting (Hayes and Games 2008). Projects such as Wilson, Connolly, Hainey and Moffat (2011) show how Scratch can be used with young children aged 8/9 years old to learn programming concepts through the introduction of game making. While Baytak and Land (2011) have focused on what has been created with Scratch during a science project undertaken with 10 to 11 year olds. Other projects focus on children using Scratch in a workshop for 13-14 year olds (Sivilotti and Laugel, 2008) or at summer camps (Adams, 2010) or computer clubhouses (Maloney *et al.*, 2008) where children enjoyed creating multimedia projects. The work done in these studies show how children use Scratch (Maloney et al., 2008) and also gain their opinions of Scratch after they have completed a workshop (Sivilotti and Laugel, 2008). Scratch has also been used at Harvard University as an introduction to programming for university students (Malan and Leitner, 2007; Malan, 2010).

3 Methods

3.1 Lesson Overview

Over the course of eight weeks, as part of their ICT lessons children were given a one hour lesson with Scratch to make games with the principle investigator leading the lessons alongside the class teacher. Table 1 shows a breakdown of the lesson plan. The final product of the lessons was a game which each grouping had created either as an adaption of the maze game or a new original game which then had to be coded to gauge the programming concepts used by the children.

Table 1: Lessons given in class

	Lesson	**Overview of Lesson(s)**
Week 1	Introduction to Scratch using Scratch cards (Rusk, 2009a)	Children were given one of the twelve Scratch cards to work on. Once completed they were able to change cards and work their way through the set.
Weeks 2-4	Building a simple maze game (Brennan, 2009) with a timer	The children were given instructions in how to create a basic maze game with one sprite and one background. They were shown how to control the sprite through the maze and then also shown how to add a timer to their game to increase challenge.
Weeks 5-8	Creating their own game in Scratch	Children were able to continue their game making either by adapting the maze game they had been working on during the previous weeks or by creating a new game by themselves.

3.2 Data coding and analysis

A coding scheme was adapted from the scheme created by Denner, Werner and Ortiz (2011) and refined based on the programming concepts that could be learned with Scratch (Rusk, 2009b). As shown in Table 2 each game was coded within three main categories – programming concepts, code organisation and designing for usability – and 22 subcategories. Each game was coded for the presence of each element (either 0/1) or in some cases the extent to which that element was used

within the categories using a range from either 0-2 or 0-3. The 29 games created were coded using this scheme.

Table 2: Game coding categories and definitions

Programming concepts found in Scratch		**Coding**
1. Sequence	Are the blocks in a systematic order to execute the program correctly?	0/1
2. Iteration	Using forever and repeat to create iterations.	0-3
3. Variables	Variables can be created within Scratch and then be used within programs.	0-3
4. Conditional Statements	Using if, forever if and if-else to check for conditions.	0-3
5. Lists (arrays)	Allows for storing and accessing lists of strings and numbers.	0/1
6. Event handling	Responding to events triggered by either the user or another script.	0-2
7. Threads	Launching two independent scripts at the same time to execute in parallel.	0-2
8. Coordination and Synchronisation	Using blocks such as wait, broadcast and when I receive to coordinate the actions of multiple sprites.	0-3
9. Keyboard Input	Using blocks such as ask and wait prompts users to type in an answer.	0-2
10. Random Numbers	Pick Random is used to select random integers within any given range.	0/1
11. Boolean Logic	Using and, or, not.	0/1
12. Dynamic Interaction	Using mouse x or y and loudness can be used as dynamic input for interaction.	0/1
13. User Interface Design	Using when sprite clicked button can create an interactive user interface.	0/1
Code organisation		
14. Extraneous blocks	Are there any blocks which are not initialised when the program is run?	-1/0
15. Sprite names (the default is overridden).	Are the default sprite names overridden?	0/1
16. Variable names	Are the variables given meaningful names when set up?	0/1
Designing for usability		
17. Functionality	Does the game run when it is started (most games start when the green flag is clicked)?	0-3
18. Goal	Is there a clear defined goal to the game?	0-2
19. Sprite customisation	Is the sprite used a predefined sprite or has the sprite been customised and to what extent.	0-3
20. Stage customisation	Is the stage used a predefined stage or has the stage been customised and to what extent.	0-3
21. Instructions clear	Has the student defined how the game is supposed to run?	0-3
22. Game originality	Students were asked to create a maze game to give them the grounding in basic skills that were required. However when it came to creating their own game students were able to adapt the maze game or create a new game entirely.	0-3

3.2 Participants

60 children (27 girls and 33 boys) aged between 8 and 11 participated in the study. They were from three classes from Royston Primary School, Glasgow. Primary 4 had 18 children; primary 5/6 had 20 children and primary 6/7 22 children. The children worked in the same pairs throughout the project: 7 pairs of boys, 5 pairs of girls, one group of 3 boys, one group of two boys and a girl and 16 boy/girl pairs.

4 Results

4.1 Game types

There were 29 games created by the groups during the study. Table 3 shows the types of games created by each class. Each game was scored in all coding categories and an overall percentage given. The mean game score was 48%. For each class the mean score was 49% (primary 4), 51% (primary 5/6) and 45% (primary 6/7) Kruskal-Wallis one way analysis of variance test showed no significant difference in game scores between class groups ($\chi^2 = 0.072$, $p < 0.965$) or between gender groupings ($\chi^2 = 0.483$, $p < 0.785$).

Table 3: Game types by class

Class	Stick with maze game	Adapt maze game (change background adapt game)	Adapt maze game (change background adapt game to two player)	Create new game (come up with another idea other than maze game)
P4	0	3	4	2
P5/6	3	1	0	6
P6/7	2	2	0	6

4.2 Concepts within games

The games developed by the students varied in their complexity, with 90% of games using keyboard or mouse control to play the game. The games that did not use event handling either did not have any scripts attached to the sprite or they used keyboard input and had the user answering questions to progress in the game. While 72% of games contained coordination and synchronisation most of these used the wait block; only one game implemented a broadcasting message to synchronise between sprites which was not fully implemented correctly as it was attached to the wrong sprites. Conditional statements within games consisted of when the sprite was moving around if it touched an object then it would cause an event to happen e.g. sprites bouncing of walls or changing scores. Table 4 shows the percentage of games that includes each programming element and only elements that were found in games are included in the table. Kruskal-Wallis tests showed no significant difference in concepts used between class groups ($\chi^2 = 0.176$, $p < 0.916$) or between gender groupings ($\chi^2 = 0.472$, $p < 0.790$). However, Mann-Whitney *U* test showed a significant difference between the concepts used in maze-based games compared to the original games made by all classes ($Z=-2.535$, $p < 0.010$).

Table 4: Concepts found in children's games

Programming Concepts	% of games including programming concepts
Sequence	93
Event Handling	90
Conditional Statements	86
Threads	83
Variables	72
Coordination and Synchronisation	72
Iteration	55
Keyboard Input	7
Random Numbers	3

4.3 Code organisation within games

Table 5 shows Code organisation within the games. 21% of the games included extraneous blocks. 72% of the games included meaningful variable names most games only included a timer, however some games also implemented a scoring system as well and only 3% of games had changed the default sprite name.

Table 5: Code organisation

Code organisation	% of games
Extraneous blocks	21
Sprite names	3
Variable names	72

4.4 Design within games

Table 6 shows the Designing for usability out of the 29 games only one had no functionality at all. All of other games had varying degrees of functionality with 28% of games being completely functional. Kruskal-Wallis tests showed no significant difference in design between class groups ($\chi^2 = 2.189$, $p < 0.335$) or between gender groupings ($\chi^2 = 0.304$, $p < 0.859$).

Table 6: Designing for usability

Designing for Usability	% of games
Functionality	97
Sprite customisation	97
Stage customisation	93
Clear Instructions	86
Game originality	83
Goal	59

4.4 In depth game examples

To illustrate the findings we present an in-depth discussion of two of the games that had the highest overall score.

4.4.1 Game 1

The first game is a fully functional maze game created by a boy/girl pair from the primary 5/6 class (Figure 1). This was an original game design with the children creating their own backgrounds and a simple shaped sprite. The object of the game is to get the sprite from the start point in the top left hand corner through the maze to the end red point before the timer runs out. This game demonstrates good use of the programming concepts. The sprite in the game is controlled by the arrow keys, which are all programmed correctly (Figure 2) and uses conditions such as: if the green walls are touched the pink sprite will bounce off them and also when the red end point is touched it signals the end of the game (see Figure 2). This game has one variable initialised within it as the timer illustrates (see Figure 2). If the end point is reached before the timer runs out then the game is ended with a sound playing and message stating “you win”, however if not then the timer ends the game with a “good try play again”. Although the timing itself has a minor fault in that the calculations used are not correct and stops when it reaches -5 rather than 0.

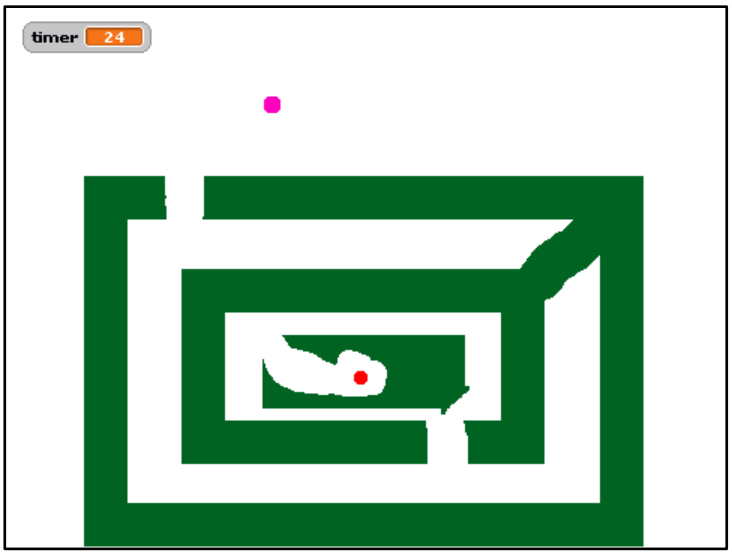

Figure 1: pair 14's original maze game

Figure 2 Game controls, conditionals and variables used in the game.

4.4.2 Game 2

The second game is from a group of boys in the primary 4 class. Their game was based on the maze game, however the background was re-designed (see Figure 3) and it was adapted into a two player game. The object of the game is to get the sprite from the start point in the top left hand corner through the maze to the end orange square on the upper right hand side, before the timer runs out. This game demonstrates good use of the programming concepts. The sprites in the game are controlled by the arrow keys for player 1 and player 2 used the w,a,s,d keys, which are all programmed correctly (see Figure 4). The game uses conditions such as If the red shapes are touched the sprites will bounce off them and also when the orange end point is touched it signals the end of the game. This game has two variables initialised within it a timer and a scoring system. While the timer works correctly – the players have 30 seconds to complete their game the scoring system needs changed as it runs similar to the timer.

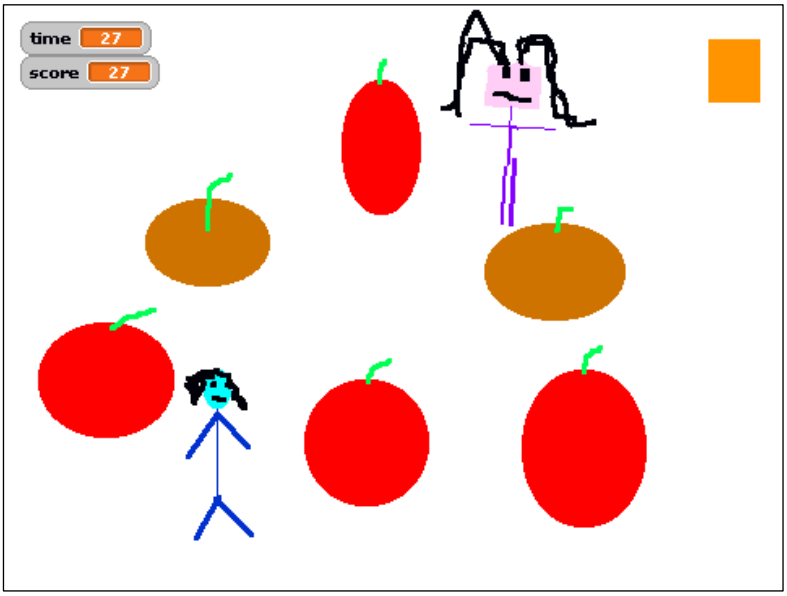

Figure 3: Group 5s two player maze game

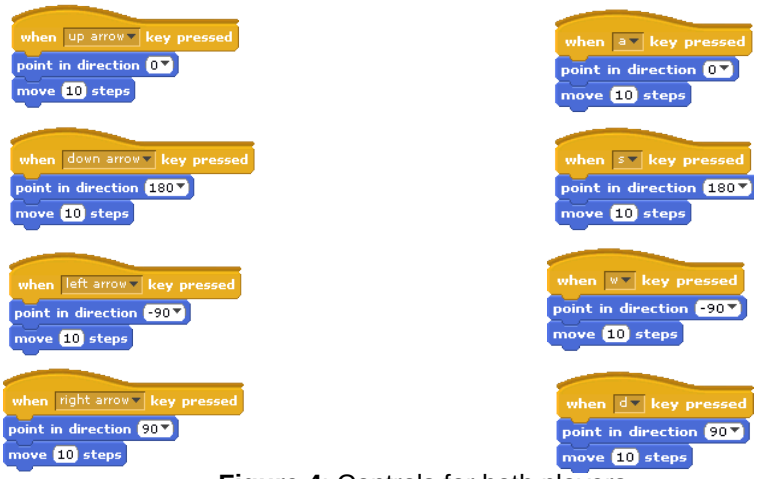

Figure 4: Controls for both players

5 Discussion

There are few prior studies that look at the learning of computing concepts though game construction. While previous research has shown that children are able to learn basic programming concepts though game making projects (Denner, Werner and Ortiz, 2010; Maloney *et al*., 2008) little is known within the classroom setting (Wilson, Connolly, Hainey and Moffat, 2011).

The eight lessons covered the basics of game making with Scratch, given the timeframe and age of children this was a basic introduction. If more time had been available then the children would have been able to progress to making more complex games and learning more programming concepts. Most groups were successful in their attempts at creating their game whether it was a maze game or original creation. The primary 4 class did not make as many original games as the other two classes, however, they preferred to adapt the maze game that they had created and had the most amount of games that were functional or only with minor mistakes. Both primary 5/6 and primary 6/7 classes had equal amounts of original games, indeed 60% of each class made their own game. However the primary 5/6 class were more successful in implementing their games with more functionality than the primary 6/7 class.

The most used programming concepts used by children in the project were similar to those found in Maloney *et al* (2008), namely User Interaction (key handling), Loops (iteration) and Conditional Statements. The use of random numbers were the lowest used programming concept.
While gender groupings did not have a significant effect on game scores the primary 5/6 class did have the highest mean score as well as the most functional games. This class consisted of mixed gender groupings.

Overall the children did manage to gain some programming concepts over the eight week period. The study has shown that even within eight hours of lessons children were able to make progress with Scratch and their learning of programming, which was similar to the results of Baytak and Land (2011), who after 10 Scratch lessons or 6 for their experienced children had completed games. As well as being similar to the game construction work of and Robertson and Howells (2008) with the Neverwinter nights project.

6 Further work

This paper has presented some important required empirical evidence in relation to the introduction of a computer game construction tool into PE level to address the issue of the dearth of empirical evidence in the GBL literature. It has also presented a coding scheme for Scratch Games. Further research will entail expansion of the study to include further primary school institutions in the Glasgow region. This will involve inclusion of different age groups to attain further empirical results to produce more statistically significant evidence and assist in refining the instruments of evaluation through a series of pilot studies. The age groups targeted would be in the age range of eight to eleven at primary four, five, six and seven level. This will enable comparisons between the different primary school levels to ascertain the suitability of the computer game construction tool at different primary educational levels. This study has particularly focused on the Glasgow region of Scotland and further comparative analysis of results attained from different regions of Scotland to assess if the results are consistent will be performed. This will produce further empirical evidence in the GBL field and address if there are differences in suitability of this approach for different Scottish regions.

Acknowledgements

This paper is a scientific publication as part of the Games and Learning Alliance (GaLA). GaLA is a Network of Excellence on 'serious games' funded by the European Union in FP7 – IST ICT, Technology Enhanced Learning (see http://www.galanoe.eu). GaLA gathers the cutting-edge European research and development organizations on 'serious games', involving 31 partners from 14 countries. The University of the West of Scotland is one of the partners.

References

Adams, J.C. (2010). Scratching middle schoolers' creative itch. In *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, pp. 356–360.

Barrios-Aranibar, D., Gurgel, V., Santos, M., Araujo, G.R., Roza, V.C., Nascimento, R.A., Silva, A.F., Silva, A.R.S. and Goncalves, L.M.G. (2006). RoboEduc: A software for teaching robotics to technological excluded children using LEGO prototypes, *3rd IEEE Latin American Robotics Symposium, LARS'06, October 26, 2006 - October 27* Inst. of Elec. and Elec. Eng. Computer Society, Santiago, Chile, pp. 193.

Baytak, A. and Land, S. (2011). An investigation of the artifacts and process of constructing computers games about environmental science in a fifth grade classroom Educational Technology Research and Development Springer Boston Vol 59 issue 7 pp 765-782

Borghi, L., De Ambrosis, A. and Massara, C.I. (1991). Logo programming and experiments to study motion in primary school, *Computers & Education,* vol. 17, no. 3, pp. 203-211.

Brennan, K**.** (2009). Maze Game, Retrieved 5 September 2011 from: http://scratched.media.mit.edu/resources/lets-play

Denner, Werner and Ortiz (2011). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts?, Computers & Education, Volume 58, Issue 1, January 2012, 240-249,

Gibson, J.P. (2003). *A noughts and crosses Java applet to teach programming to primary school children*, Computer Science Press, Inc., Kilkenny City, Ireland.

Hayes E.R. and Games I.A. (2008). Making computer games and design thinking: a review of current software and strategies Games and Culture, 3 (3–4) (July 2008), 309–332

HMIE (2009). Improving Scottish Education, HMIE, Scotland.

Kafai, Y. B. (2006). Playing and Making Games for Learning: Instructionist and Constructionist Perspectives for Game Studies. Games and Culture 1: 36-40.

Kahn, K. (1996). ToonTalk™ — An Animated Programming Environment for Children. *Journal of Visual Languages and Computing*, Vol. 7, 197 - 217.

Kahn, K. (2007), Should LOGO keep going forward 1?, *Informatics in education,* vol. 6, no. 2, pp. 307-320. Kelleher, C. and Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of

programming environments and languages for novice programmers, *ACM Comput.Surv.,*vol.37, no.2, 83-137.

Kelleher, C., Pausch, R. and Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming, ACM, San Jose, California, USA.

LTS (2011a). Approaches to learning [Online]. Retrieved 28 January 2012 from: http://www.ltscotland.org.uk/learningteachingandassessment/approaches/index.asp

LTS (2011b). The Consolarium [Online]. Retrieved 28 January 2012 from: http://www.ltscotland.org.uk/usingglowandict/gamesbasedlearning/consolarium.asp

Lindh, J. and Holgersson, T. (2007). Does lego training stimulate pupils' ability to solve logical problems?, *Computers & Education,* vol. 49, no. 4, 1097-1111

Malan, D. J. (2010). Reinventing CS50. In *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 152–156.

Malan, D. J. and Leitner, H. H. (2007). Scratch for budding computer scientists. In *Proceedings of the 38th SIGCSE technical symposium on Computer science education*. ACM, 223–227.

Maloney, J. H., Peppler, K., Kafai, Y.B., Resnick, M. and Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. In *Proceedings of the 39th SIGCSE technical symposium on Computer science education*. ACM, 367–371.

McNerney, T.S. (2004). From turtles to Tangible Programming Bricks: explorations in physical language design, *Personal Ubiquitous Comput.,* vol. 8, no. 5, 326-337.

MIT (2011). *Statistics on scratch users*. Retrieved 5 May 2011 from: http://stats.scratch.mit.edu/

Morgado, L. and Kahn, K. (2008), Towards a specification of the ToonTalk language, *Journal of Visual Languages & Computing,* vol. 19, no. 5, pp. 574-597

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas,* Basic Books, Inc.

Papert, S. (1991). Situating Constructionism. In I. Harel & S. Papert (Eds.) Constructionism Norwood, NJ, Ablex Publishing.

Resnick, M., Flanagan, M., Kelleher, C., MacLaurin, M., Ohshima, Y., Perlin, K. and Torres, R. (2009). Growing up programming: democratizing the creation of dynamic, interactive media, ACM, Boston, MA, USA.

Resnick, M., Kafai, Y. and Maeda, J. (2003). *A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantage Communities.*, United States.

Robertson, J. and Good, J. (2005). Adventure Author: An Authoring Tool for 3D Virtual Reality Story Construction. In *Proceedings of the AIED-05 Workshop on Narrative Learning Environments*, 63-69.

Robertson, J. and Howells, C. (2008). Computer game design: Opportunities for successful learning, *Computers and Education.,* Vol. 50, No. 2, 559-578.

Rusk, N. (2009a). Scratch Cards, Retrieved 5 September 2011 from: http://scratched.media.mit.edu/resources/scratch-cards

Rusk, N. (2009b). Scratch Programming Concepts, Retrieved 5 September 2011 from: http://info.scratch.mit.edu/sites/infoscratch.media.mit.edu/files/file/ScratchProgrammingConcepts-v14.pdf

Sivilotti, P. A. G. and Laugel, S. A. (2008). Scratching the Surface of Advanced Topics in Software Engineering: A Workshop Module for Middle School Students. *Learning*, 291-295.

Van Eck, R. (2006). *Where do we go from here? Ten critical areas to guide future research in DGBL.* Workshop for the annual meeting of the Games, Learning, and Society Conference, June 15–16, Madison, WI.

Vos, N., van Der Meijden, H. and Denessen, E. (2011). Effects of constructing versus playing an educational game on student motivation and deep learning strategy use. *Computers and Education*, Vol. 56, No. 1, 127-13

Williamson, B. (2009). Computer games, schools, and young people: A report for educators on using games for learning.

Wilson, A. Connolly, T.M. Hainey, T. and Moffat, D (2011). Evaluation of Introducing Programming to Younger School Children Using a Computer Game Construction Application, In *Proceedings of 5th European Conference on Games-based Learning (ECGBL)*, 20-21 October 2011, Athens, Greece.

Wyeth, P. and Purchase, H.C. (2000). *Programming without a Computer: A New Interface for Children under Eight*, IEEE Computer Society.