Computers & Education 61 (2013) 115–132

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

SLurtles: Supporting constructionist learning in *Second Life*

Carina Girvan[*], Brendan Tangney, Timothy Savage

Centre for Research in IT in Education, School of Education and School of Computer Science and Statistics Trinity College, University of Dublin, Dublin, Ireland

A R T I C L E I N F O

Article history:
Received 23 August 2011
Received in revised form
1 August 2012
Accepted 6 August 2012

Keywords:
Constructionism
Interactive learning environments
Pedagogy
Construction
Programming
Virtual worlds

A B S T R A C T

Constructionism places an emphasis on the process of constructing shareable artefacts. Many virtual worlds, such as *Second Life*, provide learners with tools for the construction of objects and hence may facilitate in-world constructionist learning experiences. However, the construction tools available present learners with a significant barrier (or 'high-floor') for the novice to first master. To address this problem, this paper presents the design concepts, first implementation and analysis of SLurtles (programmable turtles in *Second Life*), easy-to-use, programmable construction tools for use in *Second Life*. During a pilot study 24 postgraduate learners in pairs and working at distance from one another, programmed SLurtles to create interactive installations in *Second Life* over four weeks. Open interviews were conducted, chat logs recorded and learners artefacts and reflections were collected and analysed using qualitative methods. Findings show that SLurtles provide learners with a programmable, low-floor, high-ceiling and wide-wall construction tool, which supported their construction of a wide range of complex artefacts as part of a constructionist learning experience in *Second Life*.

1. Introduction

Underpinning constructionist learning experiences is the learner's engagement with programming and the construction of personally meaningful and shareable artefacts (Hoyles, Noss, & Adamson, 2002). On first glance virtual worlds appear to provide an exciting new environment to engage in constructionist learning experiences. However the inbuilt tools of virtual worlds such as *Second Life* and *Active Worlds* present the novice with a high-floor (steep learning curve) barrier to overcome (Dickey, 2005; Sanchez, 2009).

While *Scratch for Second Life* (S4SL) provides a low-floor and high-ceiling (powerfully expressive) programming environment for the virtual world *Second Life*, it does not support the construction of artefacts. Learners must first learn how to use the relatively complex 3D object modelling tools before they can engage in exploring, testing and extending their understanding. Thus to enable constructionist approaches to learning within virtual worlds, it is first necessary to lower the barriers to engagement.

To address the need for a low-floor construction tool in virtual worlds, this paper presents the design and implementation of SLurtles (programmable turtles in *Second Life*). Borrowing design concepts from Lego and Turtle graphics, SLurtles build upon S4SL and are designed to provide learners in *Second Life* and *OpenSim* with programmable tools with which to create personally meaningful and shareable artefacts. To guide research into the use of SLurtles by learners, the following question is posed:

- Do SLurtles provide learners with a pedagogical tool with which to engage in constructionist learning in virtual worlds?

Underpinning the design of SLurtles are the concepts of a low-floor (easy to use), high-ceiling (powerfully expressive) and wide-walls (support the creation of a variety of artefacts), concepts central in the design of constructionist tools for learning. Considering Norman's (1999) distinction between designed and perceived affordances, while designed for, these affordances may not emerge when SLurtles are used by learners. While a block programmed to be persistent, will be persistent, SLurtles may not be easy to use. The complexity of constructions may be constrained. Similarly, while aiming not to limit the types of constructions which learners could make, this may not be

[*] Corresponding author. Tel.: +353 (0) 18963812.
E-mail addresses: girvanc@tcd.ie (C. Girvan), tangney@tcd.ie (B. Tangney), Tim.Savage@cs.tcd.ie (T. Savage).

0360-1315/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compedu.2012.08.005

perceived to be true by the learner. Thus, these design concepts are examined through a pilot study of SLurtles with learners, which also provides an opportunity to engage in an open exploration of learners' use of SLurtles. Thus the following sub-questions are:

- Do SLurtles provide learners with a low-floor construction tool?
- Do SLurtles provide learners with a high-ceiling construction tool?
- Do SLurtles provide learners with a wide-wall construction tool?
- How do learners use SLurtles as part of a constructionist learning experience?

To pilot the use of SLurtles by learners, an exploratory case study approach is employed. 24 learners took part in a four week constructionist learning experience in *Second Life*, as part of a postgraduate course in teaching and learning. Through qualitative analysis of participant interviews, chat logs, the artefacts they created using SLurtles and their personal reflections, SLurtles are found to provide learners with an empowering low-floor, high-ceiling and wide-wall construction tool with which learners were able to engage in the construction of personally meaningful and shareable 3D artefacts as part of a constructionist learning experience in the virtual world.

2. Background

2.1. Virtual worlds

Currently the literature lacks clear consensus as to what constitutes a virtual world but broadly, virtual worlds can be described as providing multiple users with a persistent, three-dimensional environment in which they are embodied as avatars. Through these avatars users can interact with the environment and other users, whilst co-located or at a distance. Unlike massively-multiplayer online role-playing games (MMORPGs), virtual worlds lack 'game grammar', as defined by Gee (2003), and hence have no pre-defined aims or objectives for users to pursue. In addition to these generic features some virtual worlds such as *Second Life* and *Active Worlds* support user generated content. This study focuses on *Second Life*, a common choice for third-level educators using virtual worlds (Dalgarno, Lee, Carlson, Gregory, & Tynan, 2011; Kirriemuir, 2010).

It is important to note that it is not the features of a technology that support learning (Andreas, Tsiatsos, Terzidou, & Pomportsis, 2010), rather it is how they are perceived by the individual user. When an individual considers how these features could be leveraged in an educational setting this gives rise to a set of 'perceived educational affordances' (Girvan & Savage, 2010). It is these perceived educational affordances and their potential that are of interest to educators and provide opportunities for a variety of learning experiences (Warburton, 2009).

Within virtual worlds, the simulated 3D environment, avatars and range of communication tools afford interactions, a sense of self and presence. These in turn can provide the user with: a sense of immersion within the environment; embodied social presence; and opportunities for collaborative learning (Dalgarno & Lee, 2010; Dickey, 2005; Jarmon, 2009; Minocha & Roberts, 2008; Salmon, 2009). Robertson and Kipar (2010) suggest that *Second Life* can afford flexibility in both time and location of learning. Interestingly they present flexibility as a potentially negative affordance for learning, while others present the positive outcomes of this affordance, for example real-time synchronous teaching and learning (Johnson, Corazzini, & Shaw, 2011).

The creation of persistent objects is afforded through the combination of different features of virtual worlds such as *Second Life* which support user generated content Using in-world tools, objects can be created in the environment and programmed to exhibit behaviours. These objects remain within the virtual world even when the user has logged off, due to the persistent nature of the technology.

2.2. Constructionism in virtual worlds

It has been argued that the "implementation of learning in immersive virtual worlds in higher education lacks pedagogical underpinning" (Savin-Baden, 2008, p. 151). In previous work the authors have outlined an approach to tackling this shortcoming through the alignment of the perceived educational affordances of the virtual world with the features of potentially appropriate pedagogies (Girvan & Savage, 2010). Table 1 summarises the alignment between *Second Life* and the pedagogy of constructionism.

Constructionism emphasises the role of constructing personally meaningful and shareable artefacts in order for learners to actively explore, test and extend their understanding. As part of the constructionist process, Hoyles et al. (2002) emphasise the importance of programming. The tools which afford the construction of persistent objects in *Second Life* can directly support these central features of the pedagogy. In addition the incremental *bricolage* construction process (Papert, 1991) may be supported by the flexible nature of the technology and persistence can be leveraged in order to share artefacts without requiring users to be online simultaneously.

Table 1
Alignment of the perceived educational affordances of Second Life and constructionism.

m	Second Life	
Construct personally meaningful objects Actively explore, test and extend understanding	Creation of persistent objects	Construction
Opportunity to programme		Programming
Shareable artefact		Persistence
'Invisible' technology	Immersion	
Bricolage	Flexibility	
Collaborating on constructions	Collaborative learning	
In-situ	Embodied social presence	

However, the programming and object construction tools of *Second Life* present a high-floor barrier for learners to master before they can engage in such learning. Constructionist learning experiences reported in the literature required learners to have existing knowledge in order to engage in the construction of complex artefacts (Dreher, Reiners, Dreher, & Dreher, 2009). However not all learners have these skills, as a result learning experiences in *Second Life* tend not to focus upon construction but rather on social activities in-world (Sanchez, 2007, 2009). To overcome the barriers for novices to engage in constructionist learning experiences, learners require low-floor (easy to use) tools.

2.3. Low-floor constructionist tools

Many low-floor constructionist tools stem from Logo, the seminal low-floor, high-ceiling programming language. Building on Logo, Turtle Graphics was developed, providing learners with an 'object-to-think-with', whilst engaging in Logo programming. To support engagement with the turtle character, Turtle Geometry was proposed as a computational style of geometry (Papert, 1972). Unlike Euclidian geometry, in which there is a 'point' which has no properties other than 'position', Turtle Geometry uses a 'turtle', which has a position but also a 'heading' resulting from the direction it is facing (Papert, 1972, 1980). The Turtle 'object-to-think-with' thus provides the entry point to Turtle geometry which is dependent upon both position and heading and the child's existing knowledge of their own movement (Ackerman, 2004).

Whether physical or on a screen, by issuing the 'pen down' command Turtle draws a line as it moves until the 'pen up' command is executed. By creating lines on paper or screen, learners are able to observe the movement of Turtle and reflect on the result (Papert, 1980). They may then reconsider their conceptualisation of their solution to the problem, and actively experiment by changing the programme and running it again. Thus learners are supported in their engagement with experiential learning, such as through Kolb's (1984) experiential learning cycle, through the construction of shareable artefacts.

The 'pen' commands have featured in many microworlds which have been developed since Turtle. 3D examples of constructionist tools with clear links to Logo and Turtle Graphics include *PlayLOGO 3D* and *VRMath. PlayLOGO 3D* is designed to leverage the features of videogames such as narrative to motivate young learners to engage in programming (Paliokas, Arapidis, & Mpimpitsos, 2011), while *VRMath* supports the user by providing them with a first-person perspective of the turtle character (Yeh, 2010). Using *VRMath*, Yeh (2010) explored primary school students' understanding of 3D rotation by comparison to traditional pen, paper and body movement in the physical world. He found that traditional classroom approaches led to misconceptions which could be identified and conceptually resolved by the children in the 3D environment. However while these tools provide 'pen' commands, they only allow learners to create 2D lines in the 3D environment. As such, 3D constructions are considered to require advanced skills (Yeh & Nason, 2004).

Following on from Logo and Turtle, *Scratch* (Maloney et al., 2004) provides learners with a visual programming interface in which graphical blocks are snapped together in an intuitive fashion to create programmes to implement 2D interactive games or animations. Again the 'pen' commands are available and allow the learner to create lines on the screen based on the movement of a programmed sprite. *Scratch* provides a good example of how some of these more recent 'objects-to-think-with' have begun to 'widen the walls', supporting the creation of a wide variety of artefacts depending on the interests and learning styles of the user (Resnick et al., 2009).

Each of these environments can be labelled a microworld, broadly described by Ackerman and Strohecker as "carefully crafted artificial settings for creative exploration" (1999, p. 14). Computer-based microworlds are designed to enhance the most important features of a given phenomena and remove those that might distract the learner by "muddying" the outcome (Edwards, 1998). By comparison virtual worlds neither enhance the most important features for learning nor remove the distracting ones. However, they do provide educators with control over aspects of the environment, for example whether gravity is on or off. Consequently, if an educator wishes to provide a construction environment without natural physical laws in *Second Life*, they can.

In contrast to purpose built microworlds, virtual worlds do not provide learners with low-floor construction or programming tools. Instead these tools present the novice with a steep learning curve, preventing the novice user from easily creating and programming objects. They also do not support 'bidirectionality' between the programming tools and objects. Bidirectionality is described by Hoyles et al. (2002) as an important feature of microworlds, showing the movement of the object in the code and the coding of the object in its movement.

To address the issue of high-floor programming tools in *Second Life*, *Scratch 4 Second Life* (S4SL) was designed by Rosenbaum (2008). Based on *Scratch*, S4SL provides a visual programming environment outside *Second Life*, in which graphical blocks are snapped together to create a programme. On the click of a button on the user interface, the S4SL code is complied into equivalent Linden Scripting Language (LSL) code. The user then returns to *Second Life* and pastes the script into an object created in *Second Life*. Thus, S4SL allows users to quickly add behaviours and interactivity to otherwise static objects without learning the complex c-style programming language, LSL, of *Second Life*.

While S4SL provides learners with a low-floor programming environment for *Second Life*, the 3D object modelling system used to create persistent and shareable objects in-world remains a high-floor barrier to novices. Building on from S4SL, SLurtles have been designed to provide learners with low-floor programmable tools for object creation.

3. SLurtle design

In order to create a low-floor tool for the construction of objects in *Second Life*, the affordances and constraints of S4SL, existing constructionist tools and the perceived educational affordances of *Second Life* were considered. This section examines those that influenced the development of SLurtles as low-floor, programmable construction tools for *Second Life*. In Section 3.2 the design concepts behind SLurtle and their implementation are described. Finally a short walk-through of SLurtles in use is provided in Section 3.3.

3.1. Design influences

3.1.1. Turtle graphics

As previously indicated in Section 2.3, Papert's (1980) Turtle Graphics provided learners with a low-floor, high-ceiling constructionist tool, through which they could observe the effects of their programmes. Two important design concepts that came from turtle were 'position and heading' and 'construction'.

The turtle on screen or in the room showed the learner both its position and heading (Papert, 1972, 1980). Based on this the learner knew which direction it would move when given the command 'fd' (forward). An understanding of position and heading is necessary to engage in Papert's (1972) proposed computational style of geometry which was proposed in place of Euclidian Geometry in which a 'point' only has position.

Another feature of Turtle was the 'pen' command, through which a learner could create persistent patterns on screen or on paper by issuing the command 'pen down' (to start drawing) or 'pen up' (to stop drawing) as Turtle moved. It was the construction of these patters with Turtle that would be used to engage the learner in programming.

3.1.2. *Lego®*

Lego bricks provide users with a set of simple building blocks from which complex constructions can be developed. Much like other construction tools designed as toys for children, *Lego* allows the user to engage in the *bricolage* construction of increasingly complex artefacts. however unlike *Meccano* and similar toys which require some dexterity, *Lego* has a particularly low barrier to engagement, providing modular bricks which can be connected and pulled apart easily.

In the design of SLurtles, several aspects of *Lego* were influential. The simple SLurtle blocks provide a low-floor with which to engage in construction whilst the complexity, or ceiling, of construction is very high. While there is some variety, for example in length and width, the form of blocks available is typically limited to cuboid shapes, although others are available. *Lego* also strongly supports the concept of wide-walls which can be supported within *Second Life* due to the flexible nature of the environment. Individually the simple blocks do not influence the type of artefacts that can be constructed and if a limitless box of bricks is available, users are free to create a variety of artefacts that reflect their own interests. Unlike *Lego* bricks, SLurtle blocks are not connected to each other and with physical laws turned off will remain where created by the SLurtle.

3.1.3. *Scratch for Second Life (S4SL)*

Hoyles et al. (2002) note that programming remains an essential aspect of constructionist learning experiences. Of the low-floor programming tools available for use with *Second Life*, *Scratch for Second Life* (S4SL) was identified as the most powerfully expressive (highest ceiling). S4SL (Rosenbaum, 2008) provides a low-floor, visual programming interface for *Second Life*. Graphical command blocks are selected and dragged to the scripting area where they are snapped together to create a programme (Fig. 1). The concept of modular blocks that are easily snapped together and pulled apart is evident in the design of the *Scratch* interface which S4SL was developed from. On the click of the 'Copy Linden Script' button in the user interface, the S4SL code is compiled into equivalent Linden Scripting Language (LSL) code.

The user then returns to *Second Life*, creates a default script, pastes the code and places it in an object. Fig. 2 shows a section of the compiled code in *Second Life*. The script is located within the square object highlighted in yellow (in the web version).

As part of the Logo/Turtle legacy, a selection of 'pen' blocks remain in the S4SL library. 'Pen' blocks in S4SL provide an opportunity with which to create objects in *Second Life* mirroring the 2D creation of patterns with Turtle but in a 3D environment. To achieve this Rosenbaum (2008) created a 'lineSegment' object which could be placed in an object to be programmed using S4SL. When 'pen down', followed by a movement forward occurs, an instance of the 3D 'lineSegment' appears (is drawn) within the virtual world. The location and length of the new object are determined by the location of the parent object when the 'pen down' command was issued and how far it travelled in one action. For example, Fig. 3 shows the S4SL programming environment and what happens in *Second Life* after an avatar has clicked on the object (the cube) which contains both the LSL script generated by S4SL and a 'lineSegment'.

However there are constraints which need to be addressed in order to use S4SL to be used in the constructionist learning experiences. The first is that the cube in Fig. 3, when first created, is much like the Euclid point. It can be observed to have a position but no obvious heading. When programmed to move forward it will move forward but to the learner it is not clear what direction this will be until the programme is executed. In addition, S4SL users must explicitly embed a 'lineSegment' in an object so that those objects can respond to 'pen up/down' commands, introducing a new barrier for learners. Finally the 'lineSegment' is temporal lasting only a few seconds before permanently disappearing, therefore objects are not persistent and the construction and sharing of artefacts would be extremely limited.

3.1.4. *Persistence*

As identified in Section 2.1 virtual worlds afford the creation of persistent objects which can be revisited and shared following their construction, allowing learners to explore their understanding over time and compare objects side-by-side. Persistence also supports the sharing of objects with other learners. However to leverage this perceived educational affordance, SLurtles need to be able to create persistent objects. To achieve this it was necessary to reprogramme the existing 'lineSegment' used with S4SL.

3.2. *SLurtles: design concepts and implementation*

Much like Turtle Graphics was developed to with which to engage learners in Logo programming, SLurtles have been designed to engage learners in programming through the creation of persistent 3D artefacts in the virtual world. Six design concepts which underpin SLurtles were identified from Turtle Graphics, Lego and the perceived educational affordances of *Second Life*. These are summarised in Table 2.

As shown in Fig. 4, a 3D turtle was created in *Second Life* following the Turtle tradition, to provide the learner with an understanding of both position and heading of the object creation tool. To use SLurtles to create artefacts, learners must programme them using S4SL. When the 'pen down' command is used, the subsequent move forward by the SLurtle will create a SLurtle block. Through adaptation of the 'lineSegment', each block created is persistent.

Each SLurtle can create only one type of block, for example a spheroid 0.1 m in height by 1 m in width. The block begins at the starting position of the SLurtle and the length is determined by the distance the SLurtle is programmed to move in a single step. SLurtle blocks have no texture so as not to influence the type of artefacts that could be created. To provide some variety in the type of SLurtle blocks available an initial set of 16 SLurtles were created. SLurtles create either 0.1×1 (height $\times$ width), 0.1×0.1, 0.5×0.5 or 1×1 m blocks, in either cuboid, prism, cylinder or spherical form.

The complexity of constructions and variety of artefacts should only be limited by what can be programmed in S4SL. Each SLurtle includes a persistent 'lineSegment' (SLurtle block) which responds to the S4SL 'pen' commands. Using S4SL a SLurtle can be programmed to

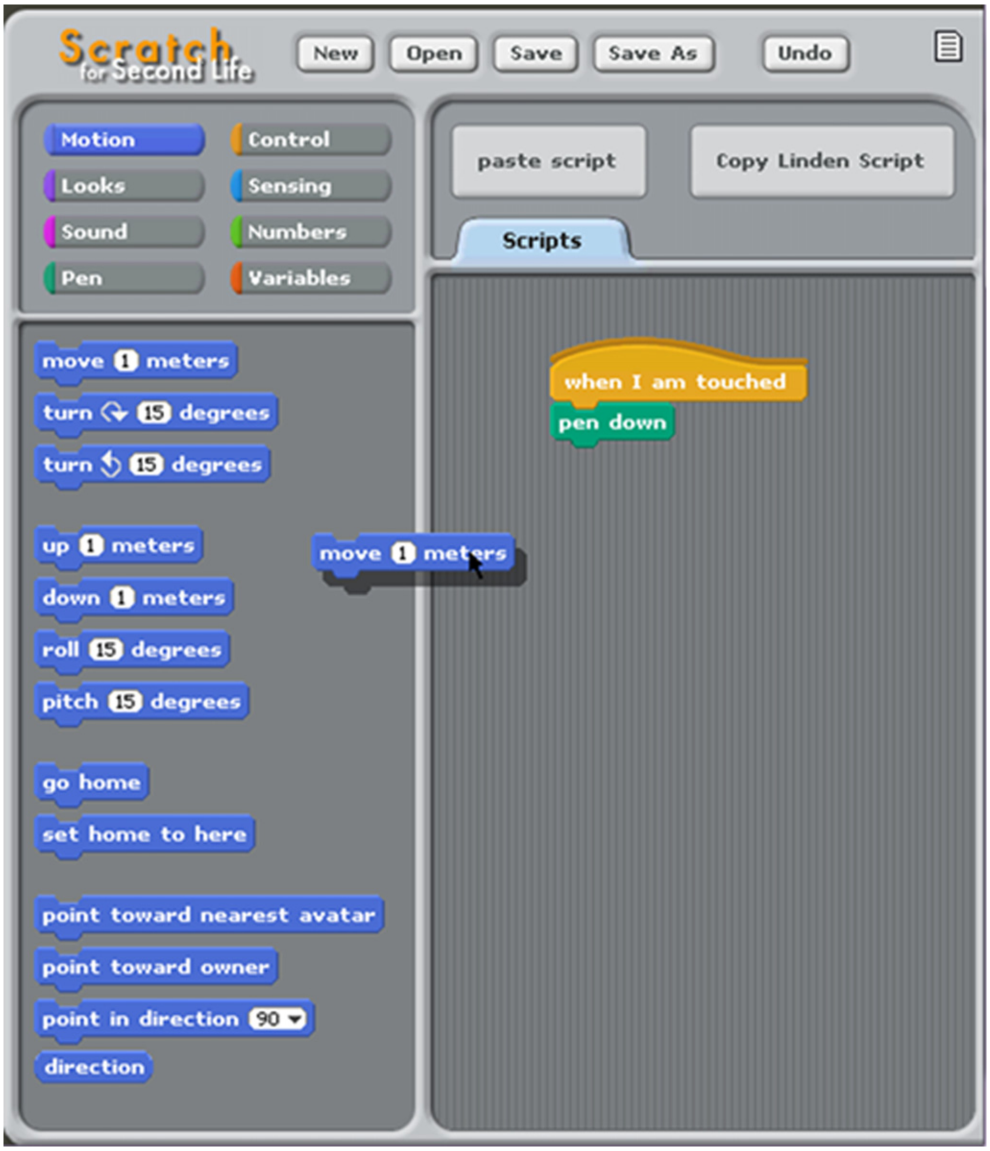


Fig. 1. S4SL application showing a block being dragged from the list of available commands to the scripting area.

move around the *Second Life* environment in three dimensions, and using the 'pen down' command can create an instance of a single SLurtle block.

3.3. *Walk-through*

SLurtles are available to learners at a 'SLurtle collection point' (Fig. 5) at which they can see a sample of each available block created with a length of 1 m. Once the learner has decided on the type of block they wish to create they click on the block and a SLurtle which will make that type of block is copied into their avatar's inventory ready for them to use.

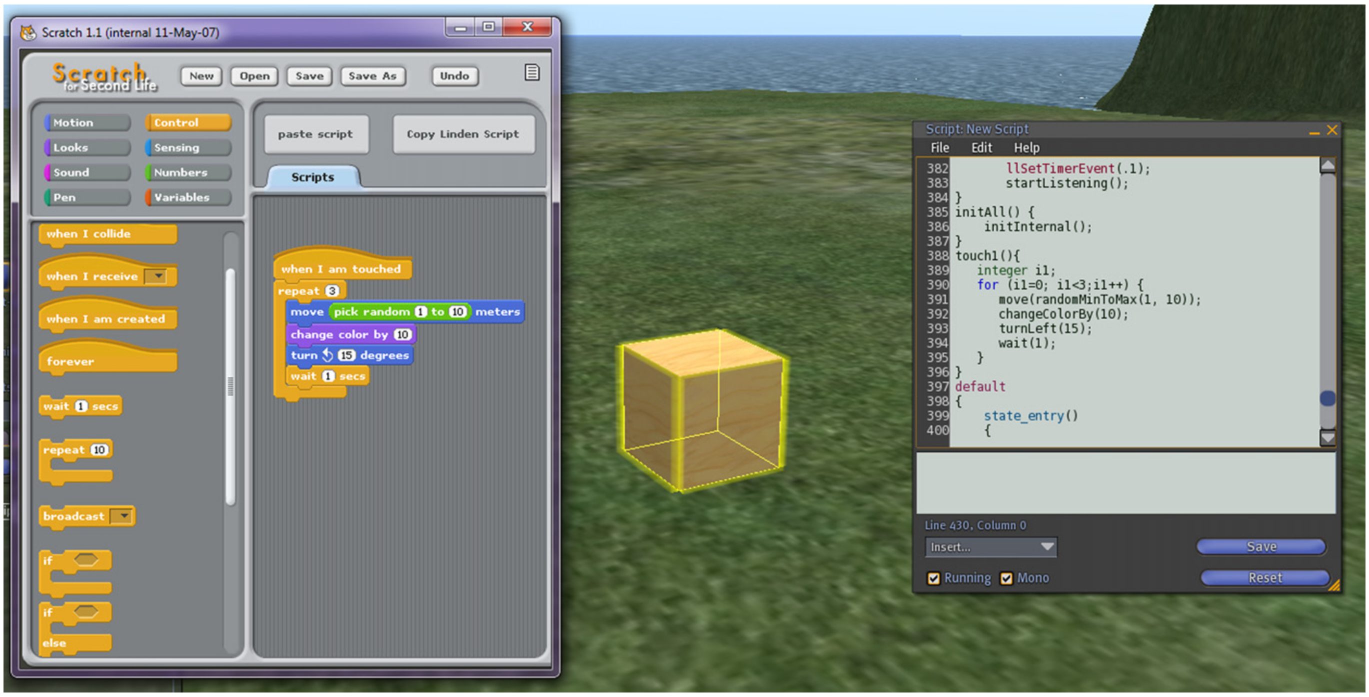
To use the SLurtle the learner takes the SLurtle from their inventory and places it in the environment. To programme the SLurtle they go to S4SL and create their code, export it via the clipboard and return to *Second Life* where they create a new script into which they paste the generated LSL code. When this script is placed in the SLurtle the SLurtle will then execute the programme (Fig. 6). Learners can then reflect on the experience before returning to S4SL to reprogramme the SLurtle.

A brief demonstration of SLurtles is provided in the video below (Video 1), recorded in *Second Life Viewer 3*.

4. Method

4.1. *Participants and activity design*

In pairs, 24 learners used SLurtles over four weeks as part of a postgraduate course in the area of technology and learning. The intended outcomes were for learners to (1) experience a constructionist learning activity and to a lesser extent (2) gain an understanding of programming. Of the 24 participants, 19 self reported as having little or no previous programming experience. Only four had experience of



```
        llSetTimerEvent(.1);
        startListening();
}
initAll() {
    initInternal();
}
touch1(){
   integer i1;
   for (i1=0; i1<3;i1++) {
      move(randomMinToMax(1, 10));
      changeColorBy(10);
      turnLeft(15);
      wait(1);
   }
}
default
{
    state_entry()
    {
```

Fig. 2. Compiled code in a script in *Second Life*.

using Second Life before the course. Prior to the learning experience, all learners participated in a general introduction to *Second Life* at distance.

The learning experience was designed to provide an opportunity to explore the use of SLurtles as low-floor construction tools as part of a constructionist learning experience. The learning experience began with face-to-face lectures and workshops on constructionism and the use of SLurtles. 12 groups were formed and where possible (in 5 of the 12 groups) non-programmers were paired with people who had some programming experience. All groups participated on the same *Second Life* island. The groups had four weeks in which to use SLurtles and S4SL to create an interactive installation in a designated 40 × 40 m space on the island (Fig. 7). They were allowed to meet face-to-face, meet

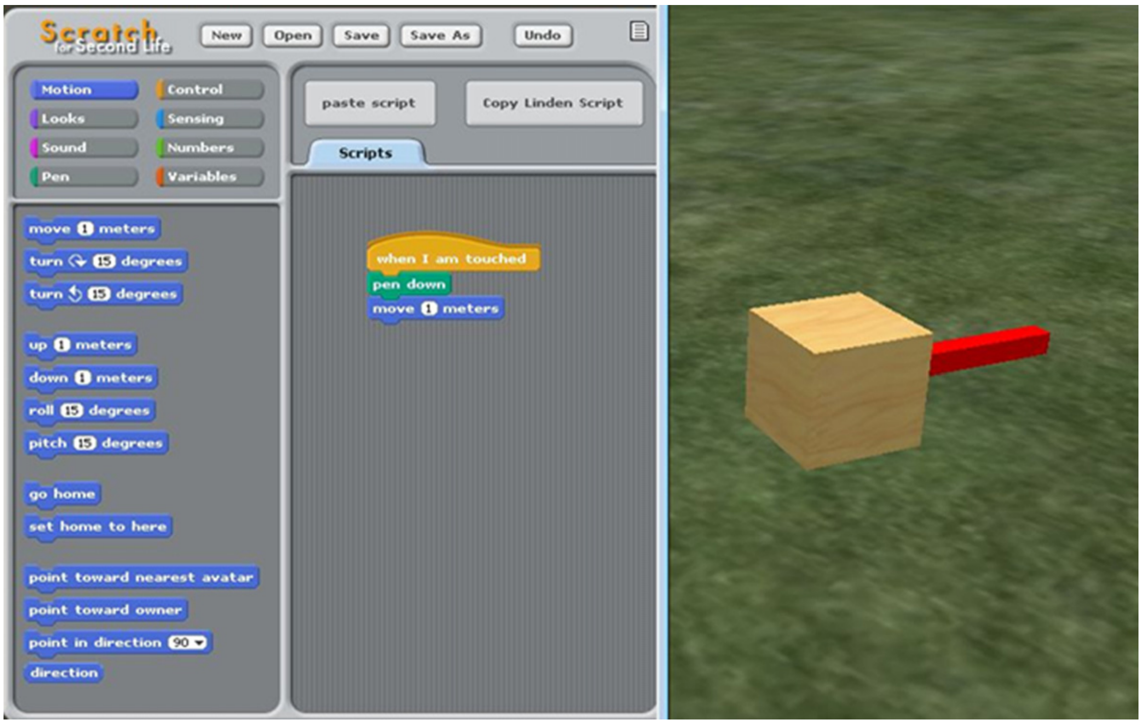


Fig. 3. S4SL programming environment on left, results of the programme in *Second Life* on right.

Table 2
SLurtle design concepts and implementation overview.

Design concept	Implementation
Provide position and heading (turtle)	Turtle character (SLurtle) used to provide position and face direction of forward motion.
Construct to engage in programming (turtle)	Use S4SL to programme SLurtle to move and create blocks using the 'pen' commands.
	An instance of the 'lineSegment' block (SLurtle blocks) is created as the SLurtle moves.
	The length of the SLurtle block is dependent on the distance the SLurtle moves in a single step.
Persistence (virtual worlds)	Adapted 'lineSegment' in each SLurtle is persistent
Simple building blocks (Lego)	Each SLurtle creates one type of cuboid, prism, cylinder or spherical object
Complex constructions possible (Lego)	Only limited by the programmes that can be created in S4SL
Variety of constructions possible (Lego)	SLurtle blocks are available in a variety of simple shapes.
	No restrictions as to how these blocks may be used.

only online or a combination of these depending on their personal preference and opportunities. At the end of the four weeks the learners were required to present their installation and reflections on the experience to the class and submit them for assessment as part of the course requirement.

4.2. Research design and procedures

In order to pilot SLurtles and answer the research questions an exploratory case study approach was implemented. The learning experience took place in a private, access controlled island within *Second Life* and as such informed consent was required from participants (Girvan & Savage, 2012). This was obtained from all 24 participants in a face-to-face setting prior to the start of the learning experience.

Interviews provide researchers with a particularly powerful research tool as a means to get 'inside a person's head' (Tuckman, 1994) in order to understand their subjective experience. As such they provide an opportunity to answer the research questions based on individuals' perceptions of their experience, following the activity. Non-directive open interviews, averaging 60 min in length, were conducted one-to-one with participants either in-world or face-to-face. Of the 24 participants, an opportunistic sample of 14 took part in the interviews, including three participants with some previous programming experience. This sample included participants only accessible to the researcher through the virtual world. While the mix of medium and location may influence the data collected, this approach was used to increase participation in the research.

While observational data can provide an accurate account of events (Stake, 1995), in this study it was particularly difficult to directly observe participants actions or the actions of their avatars in-world. This was because learners were able to access and participate in the learning experience from any location of their choosing and at any time during the four weeks. In order to gain data on the actions of participants during the learning experience, the chat logs of the learners' text-based conversations were recorded by the participants themselves. However these were limited as not all participants used the text communication tools or were able to record the chat logs.

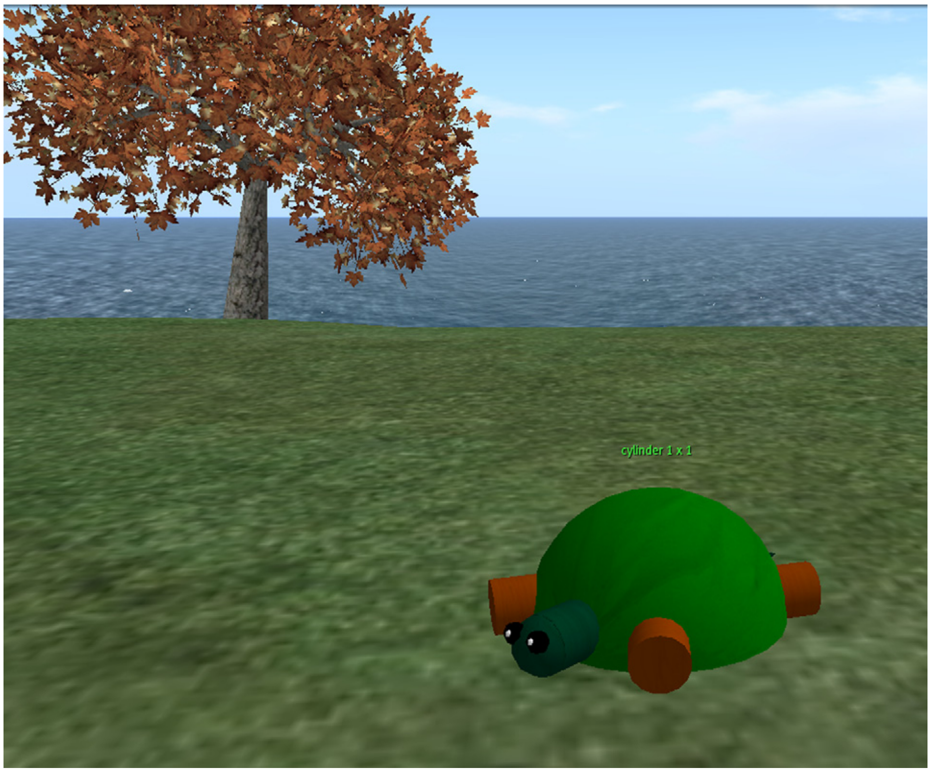


Fig. 4. A SLurtle in *Second Life*.

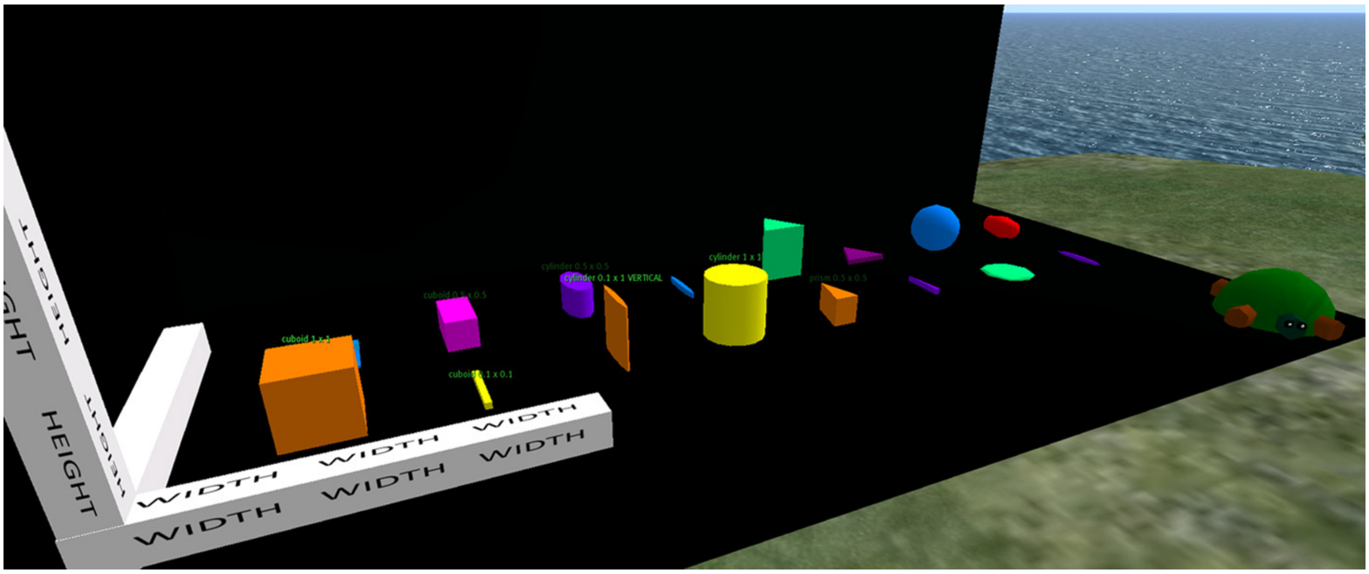
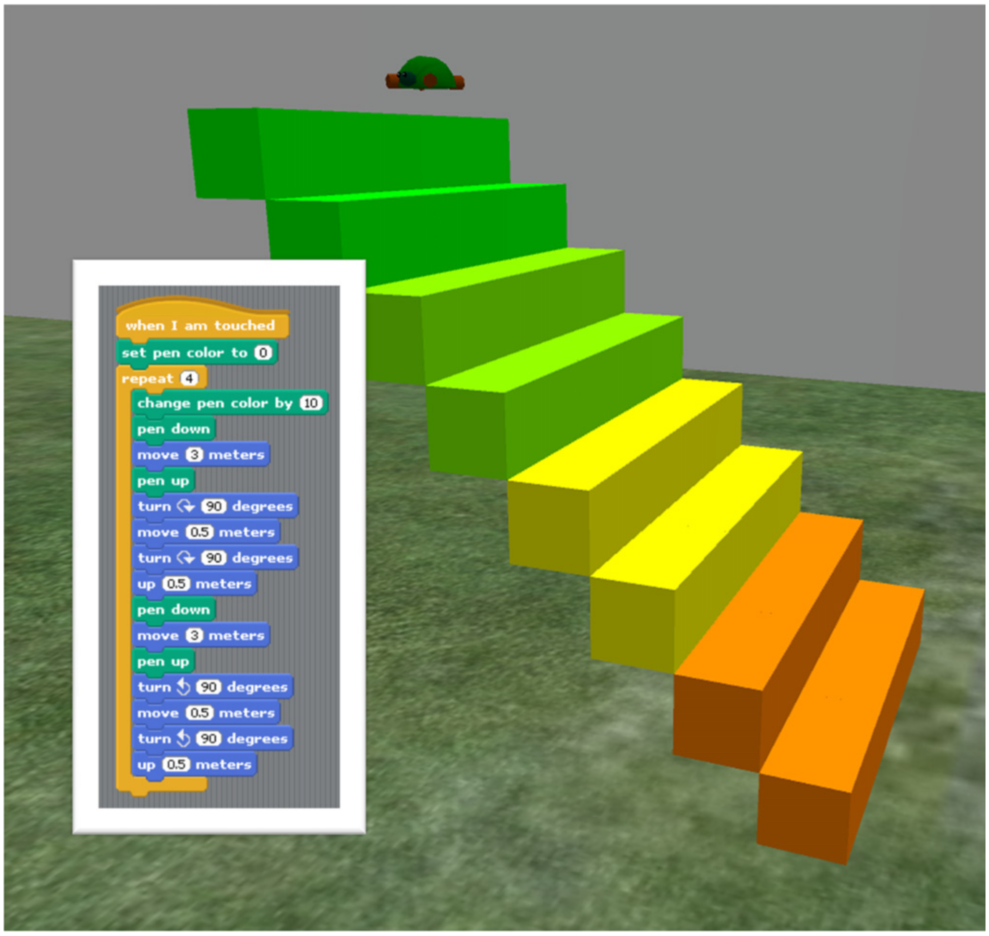


Fig. 5. SLurtle collection point.

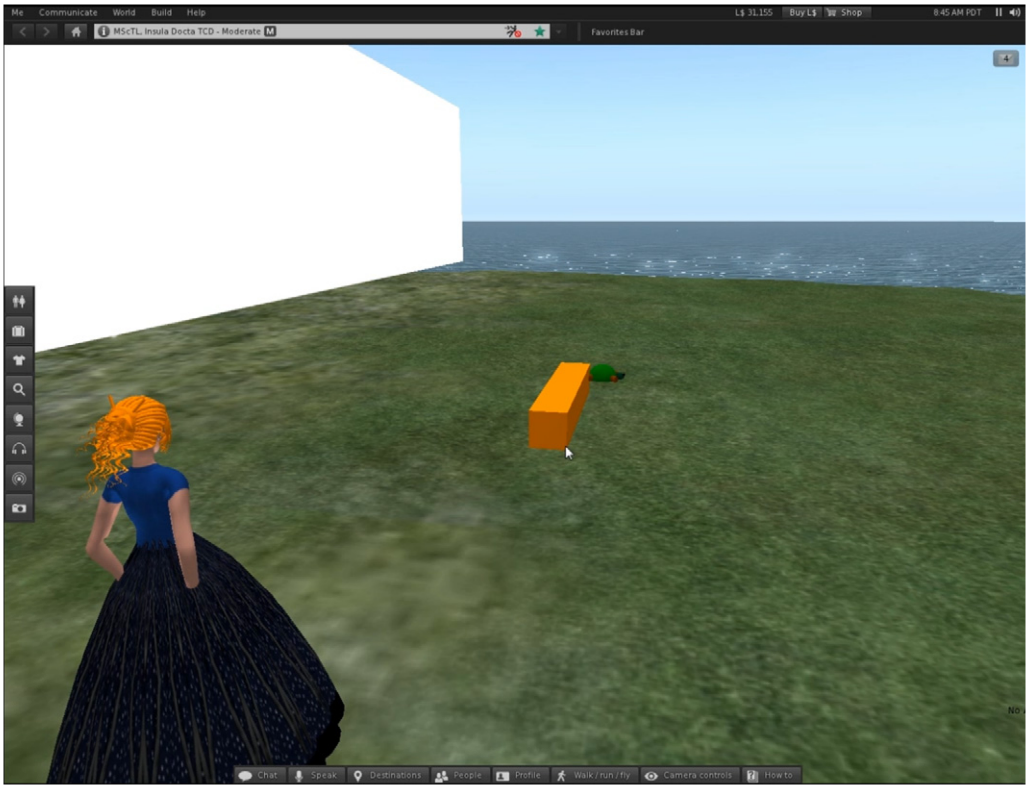
The final artefact created by each group as well as individual written reflections were also collected. These were treated as corroboratory data as they were created for a specific purpose and intended audience, which as Yin (2009) notes with regard to documents, would not be without bias.

Fig. 8 illustrates the data collection process during and following the learning experience and Table 3 presents an overview of the quantity of data recorded.



```
when I am touched
set pen color to 0
repeat 4
    change pen color by 10
    pen down
    move 3 meters
    pen up
    turn ↻ 90 degrees
    move 0.5 meters
    turn ↻ 90 degrees
    up 0.5 meters
    pen down
    move 3 meters
    pen up
    turn ↺ 90 degrees
    move 0.5 meters
    turn ↺ 90 degrees
    up 0.5 meters
```

Fig. 6. On left: S4SL code. On right: result of the SLurtle executed code.



Video 1

4.3. Data analysis

In order to answer the sub-questions the first phase of qualitative analysis followed the constant comparative approach, allowing the researcher to remain open to emergent findings. In this analysis the interviews were the primary source of data. Data analysis began with open coding of the first interview.

As Hatch (2002) notes, qualitative data analysis is a messy process. Emergent findings shaped the flow of the data analysis which included the generation and reduction of codes, formation of tentative categories, internal coding comparison and relational analysis. Rather than following a sequential approach, codes and initial categories were often returned to and re-examined (Fig. 9). For example, following the coding of the first interview tentative categories were developed, however as the number of emergent codes grew in addition to the relational analysis it became clear that these categories would not be efficient with many codes fitting in more than one category. Similarly the relational analysis highlighted possible coding properties and dimensions, requiring a return to coding. The use of researcher memos was ongoing throughout the data analysis process feeding into each stage.

Saturation of new codes appeared to be reached by the seventh interview. At this point categories and sub-categories were developed from the original codes through an iterative process which aimed to produce efficient categories (Merriam, 1998). The remaining interviews were then analysed for evidence to support or refute the categories and sub-categories. Following this, further refinement of the categories took place.

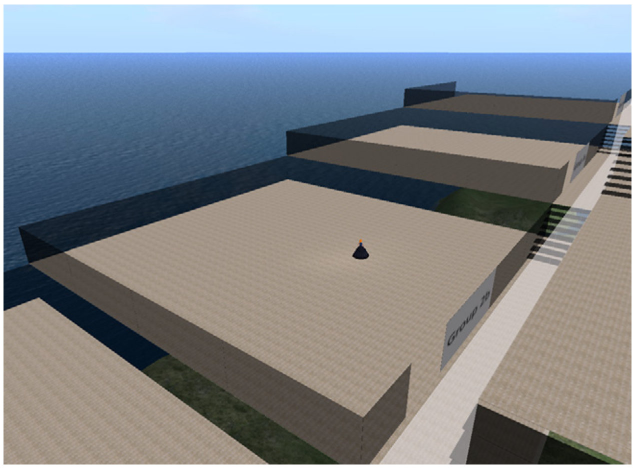


Fig. 7. Empty installation spaces with avatar prior to the activity.

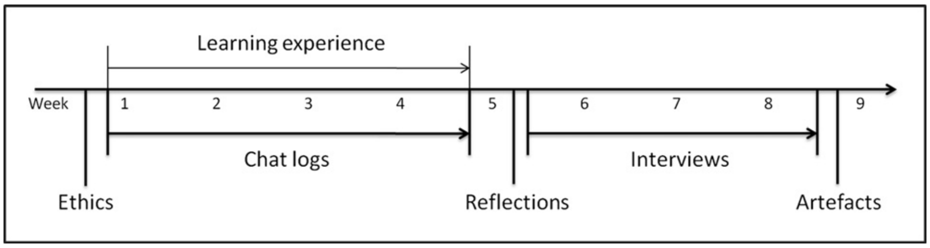


Fig. 8. Data collection process.

Table 3
Summary of data collected.

Data	Potentially available	Total collected
Interviews	24	14
Chat logs	24	5
Individual reflections	24	24
Group artefacts	12	12

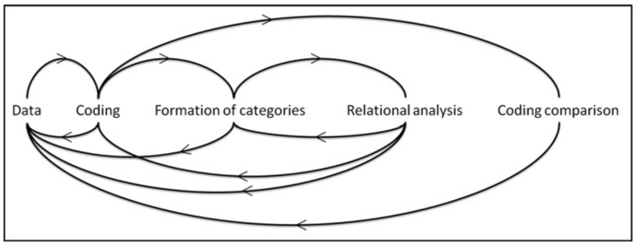


Fig. 9. The messy process of data analysis.

Data collected from learners' reflections, chat logs and in-world artefacts, provided corroboratory data used in triangulation. Following this, member checking and peer examination were conducted to increase construct validity. Higher-level analysis then aimed to develop a rich case description based on the categories and sub-categories which emerged.

The second phase of data analysis focused on the concepts of low-floor, high-ceiling and wide-walls. Separate analysis of these concepts followed the constant comparative analysis, using the researcher generated codes as an additional corroboratory data set. All data sets were analysed for both supporting and refuting evidence of the design concepts.

5. Findings

5.1. SLurtle design

Despite the initial skill barrier described above, SLurtles were described by learners as easy to use. However this did not constrain the variety or complexity of artefacts created. For example, one group with no previous programming or virtual world experience created an eclectic set of artefacts which they described as demonstrating their developing understanding of programming over the four weeks (Fig. 10).

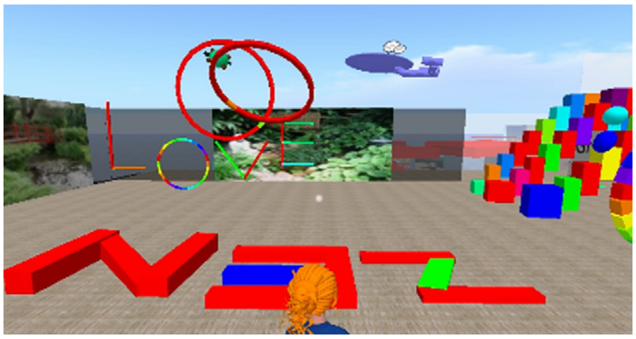


Fig. 10. Creating a variety of objects as understanding develops.

Each of the twelve groups created very different installations. For example, other installations included a piano, assault course and enchanted forest (Fig. 11). Importantly the variety of constructions was described by participants in interview and reflections as supporting learners in creating something that was of personal interest, a key feature of constructionism.

However at the beginning of the activity it was not obvious to learners what they might create. As one participant describing his groups' installation said: "*it was so obvious how to use it (SLurtles) ... but like what we could actually build wasn't obvious ... it was a lot more intricate and complicated than that and people put a lot more effort in than just, you know, doing circles or whatever*". This quote also highlights that while there were simple constructions that could be made, learners were able to explore the creation of increasingly complex artefacts. As the constructions became more complex so did the programmes used for construction.

While some artefacts created by different groups were similar there was evidence of multiple solutions to creating each object. As such the potential complexity of constructions appears to have supported the variety of constructions that were created. For example, several groups created trees as part of their installations (Fig. 12) which one participant described as requiring their most complex programme.

Although there is evidence to suggest that SLurtles provided a low-floor, high-ceiling and wide-wall programmable construction tool, there were differences of opinion amongst participants as to whether the types of blocks that could be created by SLurtles constrained the final artefacts or not. For example one participant noted that while the simple blocks supported their first experiences of building, they felt their final design was constrained by the shapes available: "*the shapes, I think for the level that we were at it did exactly what I wanted it to do. I didn't need anything any more advanced than that to get started with. Erm, so you could make pretty much…you weren't going to make something identical to what you had in your head originally, but you could make something that did look realistically like what you wanted to find*." While this suggests that the type of blocks may limit the appearance of the final artefact there was no evidence to suggest that they constrained the variety or complexity of constructions.

Similarly it was suggested that S4SL limited what could be created with SLurtles. One participant described the limited complexity of code that could be created in S4SL as limiting their final artefact. However in an interview one of the experienced programmers stated that he had not been able to reach the limits of the programming environment.

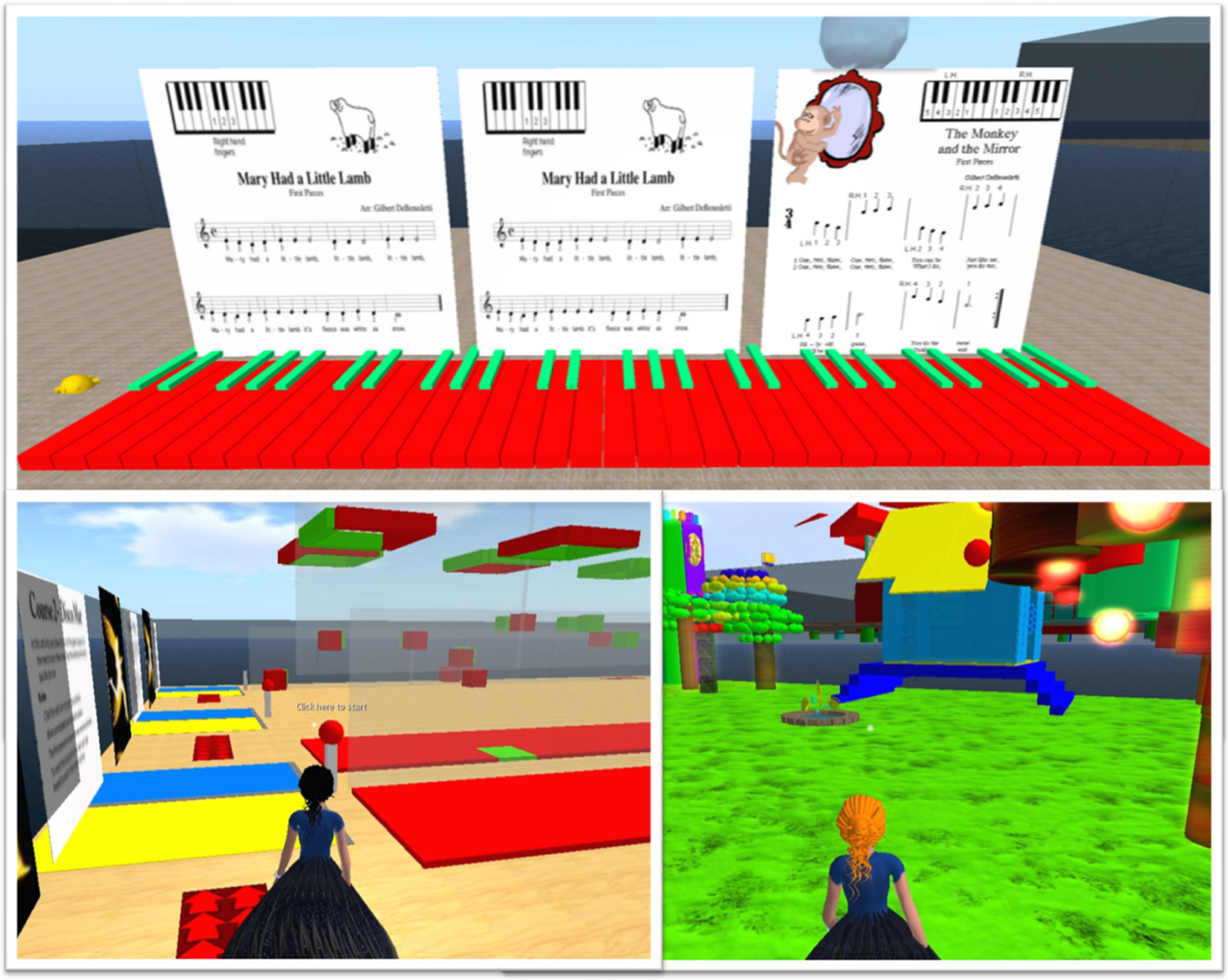


Fig. 11. Artefacts created by learners. Top: piano. Left: assault course. Right: enchanted forest.

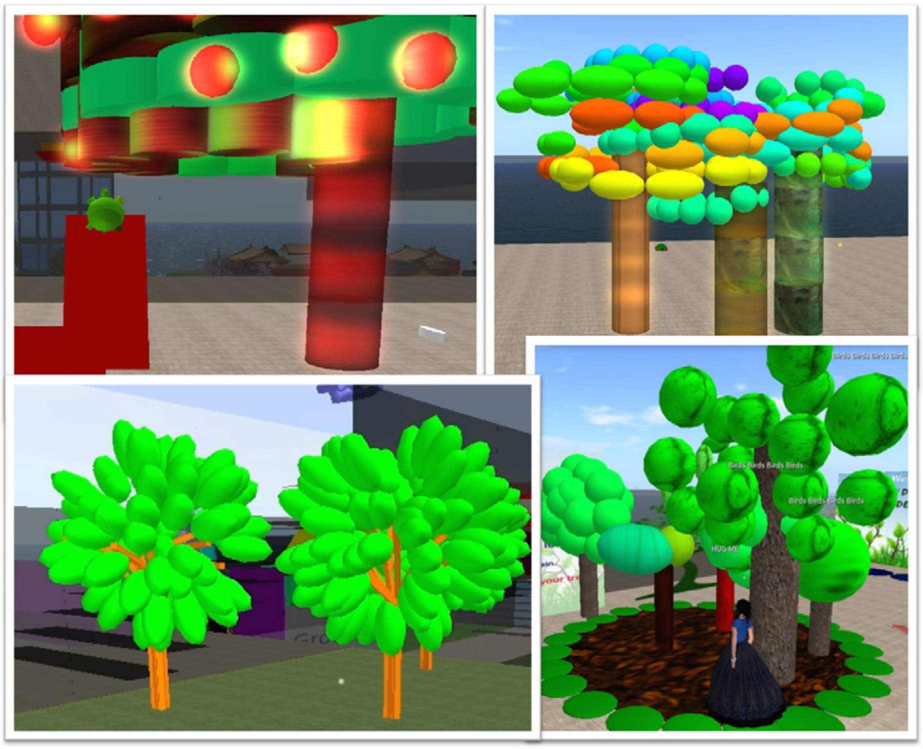


Fig. 12. Various trees created using SLurtles.

5.2. Constant comparative analysis

As an exploratory case study, the findings aim to produce a rich description of learners' use of SLurtles during the learning experience. Through the constant comparative analysis of interviews a variety of codes emerged (highlighted in **bold**) and were developed into categories. This section focuses on categories and sub-categories which provide insight into how the learners used SLurtles.

5.2.1. Thinking

Participants described SLurtles as objects within the environment which supported their thought processes and this was achieved in a number of ways. A particularly strong code, which was also supported by both chat logs and reflections, was **SLurtle movement**. For example one participant described the importance of the SLurtle showing a clear direction of movement: "*I suppose because it was something visual to look at, and you knew what way he was facing, and that was important as well when you built something*".

Several participants described SLurtles as a focal point for their thinking about programming, beginning with their initial ideas, then exploring and testing them in action with the SLurtle. As one participant said, "*getting a SLurtle to build something based on his movement. That's the key thing*". A process another participant described as "cool".

While the form of the SLurtle was important for learners, showing a clear direction of movement, one participant described the **SLurtle's appearance** as potentially constraining their thought processes: "*like why not just use a paintbrush, or you know, something that would be more relevant ... I mean, turtles don't draw*". however others described the **SLurtle's appearance** as supporting exploration and in turn creativity: "*if it's a SLurtle ... it forces you to ask, what can this thing do? and by being forced to ask, what can this thing do, you have to experiment, you have to play, and you also have to think he's that way up at the moment, could it be a different way up, and if he's this way up I can actually draw pictures on the wall*"

5.2.2. Programming

SLurtles were described in both interviews and reflections as providing learners with an opportunity to observe and reflect on the scripts that they created. As part of the **problem solving** process, SLurtle action or inaction provided learners with **feedback** on the programmes they had created "*if it didn't respond in a way that you thought that it would respond, it was providing you with feedback, you know, telling you I do not respond this way*". As noted in one reflection, this feedback supported learners as they reflected on the experience: "*The fact that I could get immediate feedback from watching the actions of the SLurtle allowed me to evaluate what the script was doing in comparison to what I wanted to happen. I found this visual feedback allowed for an accelerated understanding of what was happening within the script than if I had to think in the abstract as to what was happening*". An example of this was found in another participants' reflection which included the snapshot shown in Fig. 13. By comparison, while many learners described programming SLurtles and adding interactivity to objects as "*easy*" due to S4SL, those that added movement to objects, described the programming as "*frustrating*", as, unlike SLurtles, the objects had no obvious direction.

Interviews, chat logs and reflections provided evidence to suggest that SLurtles in the 3D environment allowed the more experienced programmers to easily explain abstract programming concepts, such as 'loops', through concrete examples to a complete novice. In

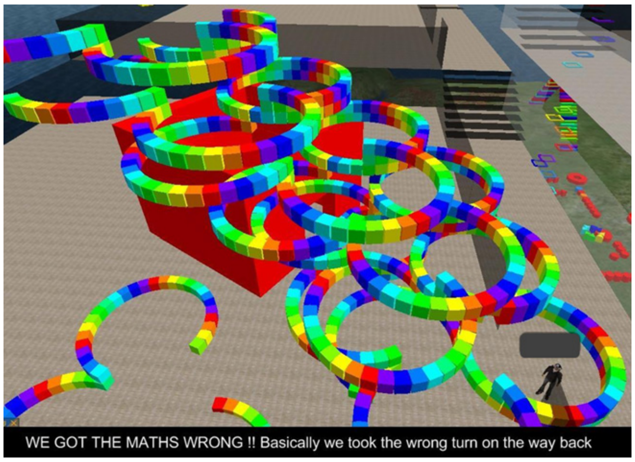

WE GOT THE MATHS WRONG !! Basically we took the wrong turn on the way back

Fig. 13. Observing programming errors.

particular it allowed the novice to watch the SLurtle perform each step within the loop before beginning the next iteration of the loop. By observing the actions of SLurtles and the creation of SLurtle blocks, learners were able to reflect on their programmes, redesign and engage in active experimentation, thus they were engaged through the process of construction. There was evidence in the chat logs to show that in some groups that had both novice and experienced programmers, the novice programmers began to adopt the language of the experienced programmers to describe the actions of the SLurtle in terms of its programme. For example, one novice programmer was recorded:

"*P1: he's gone mental*"

And some time later:

"*P1: he's obviously gone into an infinite loop*"

Another participant described SLurtles and S4SL as providing an opportunity to engage in 3D modelling and programming without existing 3D graphics or programming experience. Thus for the novice "*you're going from the concrete to the abstract. So you can basically start off doing or using*".

Although **existing knowledge** of programming was not necessary, in one reflection a participant noted that without this past experience they believed themselves to be at a disadvantage: "*I know that some of the class were able to transfer previous programming experience to this exercise and I badly lacked those terms of reference to transfer.*" Despite this, most participants did not view a lack of programming knowledge as limiting them; instead they viewed these amongst the skills they gained during the learning experience. It was also interesting to find that none of the three experienced programmers interviewed used in the in-world programming or construction tools.

In both reflections and interviews learners described SLurtles as "***engaging***" and "*fun*" objects within *Second Life* and whilst programming them was both "enjoyable" and "***frustrating***", SLurtles engaged learners in thinking about the programming concepts they were using and how to approach problem solving. as one interviewee said, "*I think it got your head around the process of programming and what programming is all about. it kind of got you into the frame of mind about the thinking that goes behind programming; that you need a lot of attention to detail and you need things exactly right otherwise they won't work.*"

5.2.3. Barriers

The sub-category of 'Barriers' was comprised of two distinctive components: 'limitations and constraints' and 'enablers'. 'Limitations' were distinct to 'constraints', as 'limitations' were perceived as preventing the learner from achieving their goals, whereas learners described working with 'constraints' to achieve their goals. In addition, what one participant described as a limitation another might describe as a constraint and therefore these codes were combined into one. 'Enablers' is used to describe low barriers or those objects, tools or processes that allowed barriers to be lowered.

5.3. Limitations and constraints

Building in *Second Life* with or without SLurtles is not without its limitations and there was evidence that learners had **expectations** of what they would and would not be able to achieve at the start of the learning experience based on their **past experiences**: "*when I first came into it I was thinking, it's just like a drawing program. You're drawing for continuous lines but you're not. and then once you figure that out, you kind of go well, actually, you can use this to your advantage ... once you get to that point you can do there's lots of other interesting things you can do that you wouldn't have thought if it had just been a line drawing program. so in a way, it not being as you expected to be was a good thing because it forced you to go in a completely different direction think in a completely different direction.*"

During their initial explorations, learners discovered that "*the SLurtle builds from the centre of the SLurtle so it is a bit difficult when you're trying to get it to turn a certain way, but it turns and it starts building from the centre of that block rather than the very side*". Whilst one participant described this as a limitation, some described it as a constraint to be worked with, a challenge to problem solve and overcome, whilst others did not identify it as a barrier.

The number of blocks (prims) that each group could create using SLurtles was restricted due to a limitation within *Second Life* on the number of prims that can be created on an island. Fig. 14 shows individual SLurtle blocks highlighted in yellow (in the web version), as part of a larger construction. In both artefacts and interviews there was evidence of learners exploring, programming concepts such as loops through the creation of objects with multiple blocks in them, however due to the **prim limit** learners often deleted these artefacts in favour of less prim-heavy constructions. As a result, "*to build a round tower, with doors and windows in it you actually need to create lots of bricks which are in a circle, which went way over the prim counter to actually do it. Yeah, there were other ways of doing it, but prim count would be the one thing that restricted you in that environment*".

Another barrier that several participants reported was the *Second Life* **permissions** system, whereby learners can share objects and the contents of scripts. This was described as particularly problematic when collaborating in the scripting of a SLurtle. Some groups were able to overcome this using the tools available to them in *Second Life* such as described by one interviewee: "*shorthand descriptions of scripts ... typed on the chat line or a texture showing an example was pulled into second life and displayed*" (illustrated in Fig. 15 taken from the same learners' reflection). However those that did not master the permissions system or an alternative approach found that, in the worst case, their collaboration turned into two people working independently on the same project.

While many participants described SLurtles as enabling them to easily engage with building in *Second Life* and described S4SL as an **easy-to-use** programming environment, "*putting the script together wasn't hard it was just knowing how to get that script and you know paste it and open it and put it into the turtle and stuff like that, just trying to remember the sequence*". This skill-based task of transferring the script from S4SL to the SLurtle was described as a barrier by two participants. With practice this was overcome, however it limited the ease with which they could initially engage with the SLurtles.

Other features of *Second Life* which participants described in their interviews, but most often in their reflections, as limiting or constraining the learning experience included the text-based chat tools, voice in *Second Life*, and skills such as movement of the avatar and **camera controls**. However some participants described that by engaging with the SLurtles they quickly overcame these barriers or they became irrelevant: "*And what was interesting about the SLurtles and engaging with your own scripting control of something was, you're suddenly focused on something inside the environment, where the environment now becomes, goes into your peripheral sort of consciousness*."

5.4. Enabling

It is worth noting that despite the barriers imposed by Second Life, learners described SLurtles as providing them with a tool without which "*we wouldn't have been able to build what we have built*". In one reflection a learner wrote about the ease at which he was able to use the SLurtles to create objects. He described feeling like "*a child with a new toy and I wanted to play. There was no need to read the instructions*." Over time learners were able to achieve a high level of control and accuracy in what they created through S4SL and SLurtles by "*just put*(ting) *in exactly what you want the SLurtle to do and it did the exact angles*". However, achieving this high level of control through their programming skills was not easy.

Learners described an initial process of exploring what they could create with SLurtles before focussing on the creation of specific objects for their assignment: "*it was very interesting ... it was a bit of fun. ... Yes, I enjoyed them when I could do random stuff with them, and I thought oh that's good, that's cool, but then when you had to start applying specific ideas, and get it to do this and that, erm, it became a bit more difficult*".

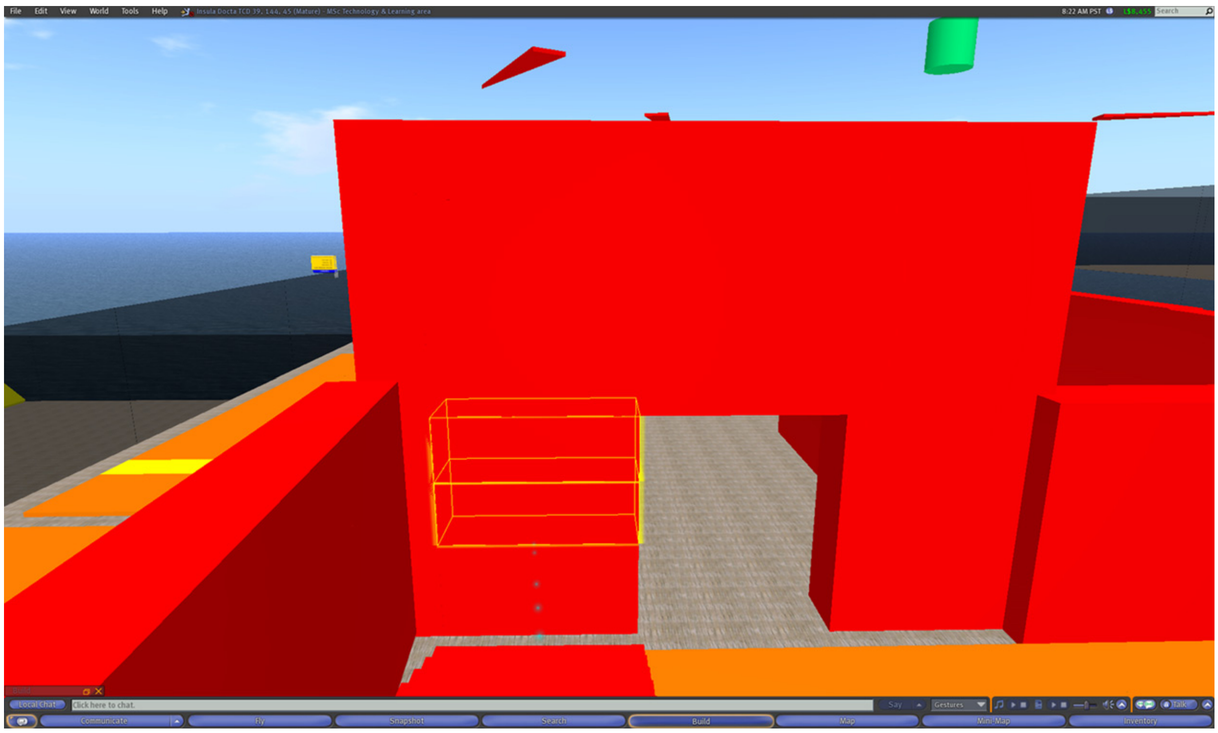
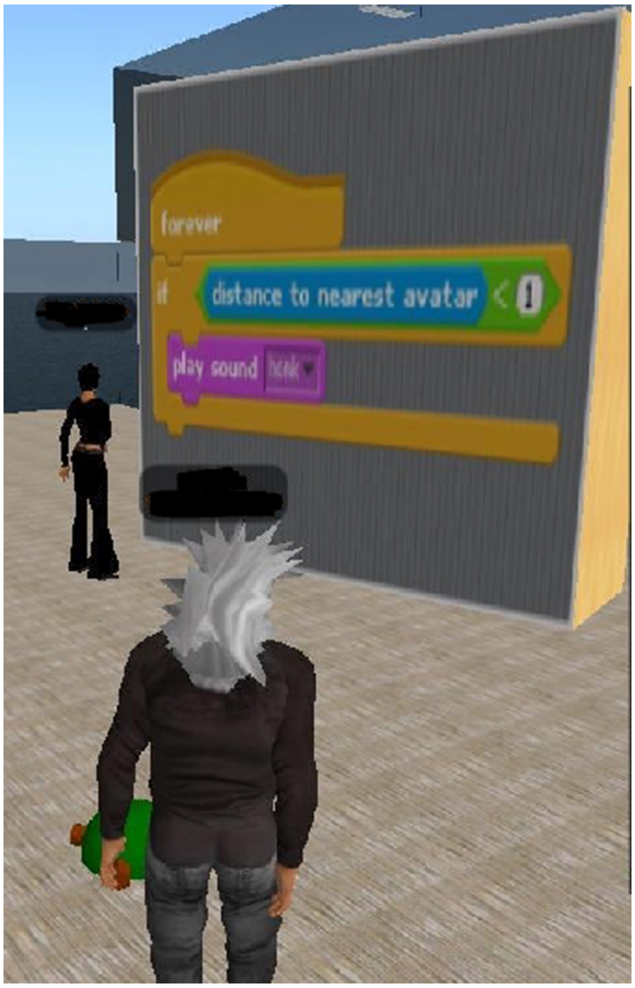


Fig. 14. Red brick house created using loops with two individual blocks highlighted in yellow. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



```
forever
  if distance to nearest avatar < 1
    play sound honk
```

[redacted]

[redacted]

Fig. 15. Using images to support collaboration.

in one reflection a learner described the lows and resulting highs “*changing one script numerous times until the SLurtle did exactly what I asked it to do. Sometimes it was frustrating but the sense of satisfaction when the SLurtle built a perfect house or a beautiful tree was worth all of the frustration*”.

6. Discussion

S4SL was created to lower the floor and allow learners to engage in programming in virtual worlds. However to engage in constructionist learning experiences, learners also require a low-floor construction tool. Following the design of SLurtles, the pilot study reported in this paper aims to answer the question:

- Do SLurtles provide learners with a pedagogical tool with which to engage in constructionist learning in virtual worlds?

In order to answer this question, four sub-questions were posed. The first sub-question focuses on whether or not SLurtles provided the learners with a low-floor construction tool, which was the initial motivation behind the construction of SLurtles. The findings demonstrate that SLurtles are easy-to-use construction tools without which learners stated they would not have been able to achieve such complex artefacts. There was, however, an initial skill barrier described by some participants when transferring a script from S4SL to SLurtle and this limited their initial engagement with SLurtles. Although there was only evidence of this from two interviews, it is unclear how many other participants may also have experienced this barrier but not reported it. While this may raise the floor for initial engagement with SLurtles, the creation of objects using the 3D object modelling tools in *Second Life* would, for novices, be a much higher barrier to entry.

Generic *Second Life* skills and tools were also identified as barriers, in particular avatar movement, camera controls, permissions and communication tools. It is unclear from the findings as to what extent these limited engagement with SLurtles, however once learners engaged in the use of SLurtles these barriers were quickly overcome or became irrelevant. These barriers suggest that there may be some in-world skills that learners need to successfully master in order to initially or fully engage with SLurtles, however there is insufficient data to draw any firm conclusions on this.

The second and third sub-questions consider the high-ceiling and wide-wall concepts of constructionist tools in relation to SLurtles. Each of the groups created very different installations, suggesting that SLurtles do support the wide-wall concept. There were mixed views on the complexity of the artefacts that could be created using SLurtles. While some considered S4SL or the type of SLurtle blocks limited what could be created, others viewed these aspects of SLurtles as enabling them to create complex artefacts.

The fourth sub-question explores the learners' use of SLurtles as part of a constructionist learning experience. The SLurtle character, its clear position and heading together with the blocks it created, provided learners with a focal point through which they were able to gain feedback and reflect on their programmes. Learners were able to quickly engage in using the SLurtles to create objects and explore and test their understanding through concrete constructions.

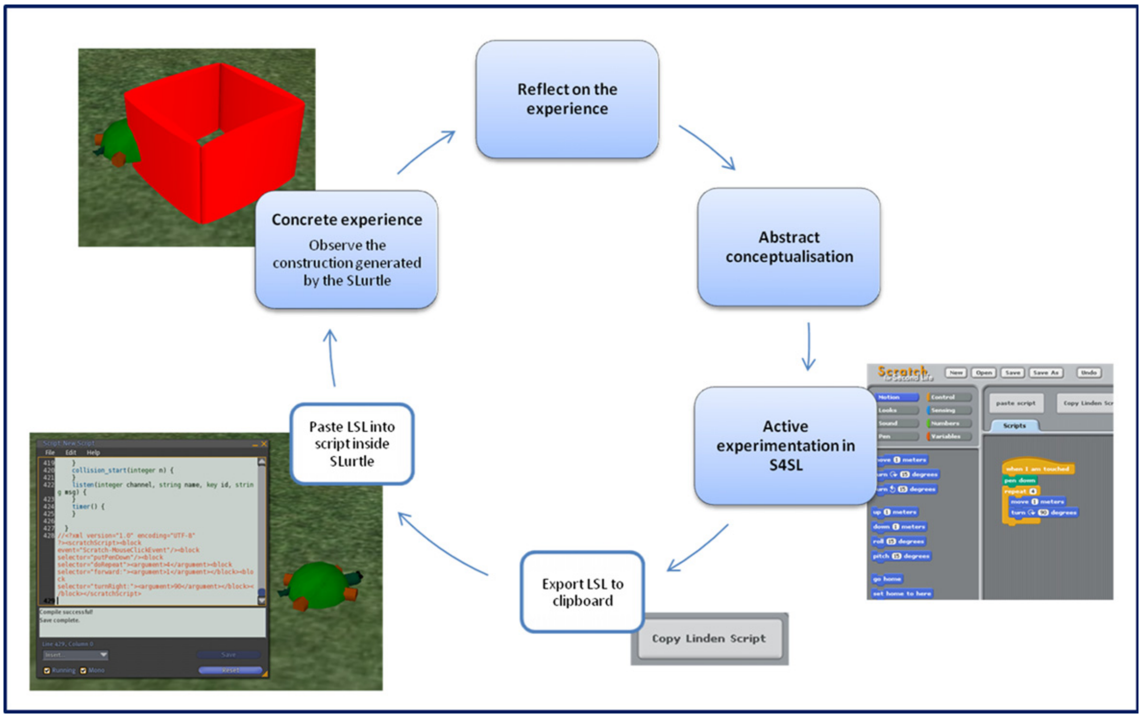
The limited variety of simple blocks was used to create a wide variety of artefacts of varying complexity, identified as personally meaningful constructions. There was evidence that as constructions became more complex so did the programmes used for the constructions. It is also interesting to note that there was no evidence of the experienced programmers using the in-world programming tool, suggesting that S4SL provided a suitably high-ceiling programming environment with which to programme SLurtles. Due to the persistent nature of the blocks, learners were able to share their artefacts during and after construction. This supported collaboration with their partner and the wider class group.

Within the category 'Thinking' there is evidence of learners engaging in an experiential learning cycle (Kolb, 1984). Although Kolb's model is often critiqued, it provides a useful frame of reference to which the findings in the category of 'thinking' can be discussed. The SLurtle's movement, indicated by its clear position and heading, as well as the SLurtle blocks created, provided the learners with a concrete experience which they could observe and reflect upon. This provided learners with an opportunity to reassess their code asking questions of themselves and their partner including "why didn't it work, or why did it work?" and comparing the outcome to the original plan. This was followed by the redesign of the programme which was exported from S4SL into *Second Life* and implemented by the SLurtle to provide the learner with a concrete experience. The full stages of this process, mapped onto Kolb's model, are illustrated in Fig. 16.

While Hoyles and Noss (2003) may argue that as the learners in this context are not engaged in a microworld they will lose the psychological connection between the abstract code and the concrete output, the findings show that the learners described using SLurtles as supporting the exploration and testing of their abstract understanding of programming through concrete constructions. However as S4SL is currently unavailable within *Second Life*, the disconnect that may result from moving between two programmes could reduce the effectiveness of SLurtles in supporting learners' concrete experience. In addition it has the potential to diminish the sense of immersion experienced by learners and the tools with which learners engage may not become 'invisible', an important feature of constructionist learning experiences (Papert, 1980s, 1991).

The process of exporting the code from S4SL into the SLurtle was also identified as an initial skill barrier and demonstrates the lack of 'bidirectionality' (Hoyles et al., 2002) between code and SLurtle. While this lack of bidirectionality also supports the notion that this learning context does not provide a microworld for learners, it is important to note that this was not the aim when developing SLurtles, but rather to create a low-floor construction tool. To address these issues, further development of S4SL to provide a representation of the S4SL environment in *Second Life* may be necessary.

Despite some limitations, SLurtles clearly supported learners' creation of artefacts as they engaged in the constructionist learning experience. While not easily achieved, learners began to explore more complex constructions over time. Exploring, testing and extending their understanding they gained a high sense of satisfaction when they were able to programme the SLurtles accurately.



```
    }
    collision_start(integer n) {
    }
    listen(integer channel, string name, key id, strin
g msg) {
    }
    timer() {
    }

  }
//<?xml version="1.0" encoding="UTF-8"
?><scratchScript><block
event="Scratch-MouseClickEvent"/><block
selector="putPenDown"/><block
selector="doRepeat"><argument>4</argument><block
selector="forward:"><argument>1</argument></block><blo
ck
selector="turnRight:"><argument>90</argument></block><
/block></scratchScript>
```

```
Compile successful!
Save complete.
```

Fig. 16. The use of SLurtles mapped onto Kolb's experiential learning model.

SLurtles engaged learners in thinking about programming by providing them with a programmable low-floor tool for the construction of a wide variety of personally meaningful and shareable artefacts within the virtual world *Second Life*. Therefore the answer to the main research question is yes, SLurtles provide learners with a pedagogical tool with which to engage in constructionist learning in virtual worlds.

7. Conclusion

The initial motivation for creating SLurtles was due to the high-floor object construction tools currently available in virtual worlds (Dickey, 2005; Sanchez, 2009) which may limit learners' engagement with constructionist learning experiences in-world. This paper has presented the design and pilot study of SLurtles as low-floor, programmable construction tools for use in the virtual world of *Second Life*.

The findings demonstrate that SLurtles provided learners with a low-floor programmable construction tool with which they were able to create a variety of complex artefacts. While initial constructions were easy, more complex constructions required more complex programmes and with this came a sense of challenge, hard fun and achievement. Although most learners had little or no previous programming or virtual world experience, they were able to collaborate at distance to create complex artefacts with SLurtles.

The intention of this study was to pilot SLurtles, considering the floors, ceilings and walls which may constrain learners. While they are shown to provide a low-floor, high-ceiling and wide-wall construction tool, much of the data is drawn from learners' perceptions. Further research is needed to understand the points at which the floor, ceiling and walls stop and begin to limit what learners may achieve. This may undertaken in comparison to other in-world tools or other constructionist environments such as *Scratch*. However the need for such research also highlights a potential limitation in the discourse surrounding constructionist tools, that is how to measure floors, ceiling and walls and whether these are comparable between tools. Within this paper it is not possible to engage in such a debate, however it is an issue which needs to be clearly addressed by the field.

While SLurtles and S4SL may under-go further development to address the concerns highlighted in Section 5 and the barriers described in Section 4, they have been found to provide a low-floor tool for the construction of objects in *Second Life*. Although designed for *Second Life*, SLurtles have the potential to be used in a number of other virtual worlds developed through the *OpenSim* project. Although presented in terms of *Second Life*, the design concepts behind SLurtles could also be implemented for the development of similar tools in other virtual worlds such as *Active Worlds*.

With a low-floor programmable construction tool, SLurtles are currently being implemented in further work which aims to develop an understanding of constructionism in virtual worlds. In particular this work explores the public nature of construction, the role of avatars and barriers to engagement.

In *Mindstorms*, Papert (1980) advocated "the construction of educationally powerful computational environments that will provide alternatives to traditional classrooms and traditional instruction" (p. 182). With SLurtles providing the low-floor programmable construction tool with which learners can engage in constructionist learning activities, virtual worlds may provide such an educationally powerful computational environment.

Role of the funding source
The funding sources have had no impact on the study, the writing of the report or the decision to submit the paper for publication.

Acknowledgements

The authors would like to thank members of the Centre for Research in IT in Education, Trinity College, University of Dublin for their support during this work as well as the reviewers for their helpful suggestions. The authors also wish to acknowledge the funding provided through the 'Embark Initiative' of the Irish Research Council for Science Engineering and Technology.

References

Ackerman, E. (2004). Constructing knowledge and transforming the world. In M. Tokoro, & L. Steels (Eds.), *A learning zone of one's own: Sharing representations and flow in collaborative learning environments* (pp. 15–37). Oxford: IOS Press.
Ackerman, E., & Strohecker, C. (1999). *Build, launch, convene: Sketches for constructive-dialogic learning environments*. MERL – a Mitsubishi Electric Research Laboratory, Retrieved 25.10.09, from. http://www.merl.com/papers/TR99-30/.
Andreas, K., Tsiatsos, T., Terzidou, T., & Pomportsis, A. (2010). Fostering collaborative learning in second life: metaphors and affordances. *Computers & Education, 55*(2), 603–615.
Dalgarno, B., & Lee, M. J. W. (2010). What are the learning affordances of 3-D virtual environments? *British Journal of Educational Technology, 41*(1), 10–32.
Dalgarno, B., Lee, M. J. W., Carlson, L., Gregory, S., & Tynan, B. (2011). An Australian and New Zealand scoping study on the use of 3D immersive virtual worlds in higher education. *Australasian Journal of Educational Technology, 27*(1), 1–15. http://www.ascilite.org.au/ajet/ajet27/dalgarno.html.
Dickey, M. D. (2005). Brave new (interactive) worlds: a review of the design affordances and constraints of two 3D virtual worlds as interactive learning environments. *Interactive Learning Environments, 13*(1–2), 121–137.
Dreher, C., Reiners, T., Dreher, N., & Dreher, H. (2009). Virtual worlds as a context suited for information systems education: discussion of pedagogical experience and curriculum design with reference to second life. *Journal of Information Systems Education, 20*(2), 211–224.
Edwards, L. D. (1998). Embodying mathematics and science: microworlds as representations. *Journal of Mathematical Behaviour, 17*(1), 53–78.
Gee, J. P. (2003). *What video games have to teach us about learning literacy*. New York: Palgrave Macmillan.
Girvan, C., & Savage, T. (2010). Identifying an appropriate pedagogy for virtual worlds: a communal constructivism case study. *Computers & Education, 55*(1), 342–349.
Girvan, C., & Savage, T. (2012). Ethical considerations for educational research in a virtual world. *Interactive Learning Environments, 20*(3), 239–251.
Hatch, J. A. (2002). *Doing qualitative research in educational settings*. Albany: State University of New York Press.
Hoyles, C., & Noss, R. (2003). What can digital technologies take from and bring to research in mathematics education? In A. J. Bishop, M. A. Clements, C. Keitel, J. Kilpatrick, & F. K. S. Leung (Eds.), *Second international handbook of mathematics education* (pp. 323–349) Dordrecht: Kluwer Academic Publishers.
Hoyles, C., Noss, R., & Adamson, R. (2002). Rethinking the microworld idea. *Journal of Educational Computing Research, 27*(1), 29–53.
Jarmon, L. (2009). Pedagogy and learning in the virtual world of second life. In P. Rogers, G. Berg, J. Boettcher, C. Howard, L. Justice, & K. Schenk (Eds.), *Encyclopaedia of distance and online learning* (2nd ed.). (pp. 1610–1619) Hershey, PA: IGI Global.
Johnson, C. M., Corazzini, K. N., & Shaw, R. (2011). Assessing the feasibility of using virtual environments in distance education. *Knowledge Management & E-Learning: An International Journal, 3*(1), 5–16.
Kirriemuir, J. (2010). Virtual world activity in UK universities and colleges (Spring 2010). Accessed on 26.09.10 from. http://virtualworldwatch.net/wordpress/wp-content/uploads/2010/05/Snapshot-8.pdf.

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development*. Englewood Cliffs, NJ: Prentice-Hall.

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004). Scratch: a sneak preview. In Y. Kambayashi, K. Tanaka, & K. Rose (Eds.), *Proceedings of the second international conference on creating, connecting, and collaborating through computing* (pp. 104–109). Kyoto: Kyoto University.

Merriam, S. B. (1998). *Qualitative research and case study applications in education*. San Francisco: Jossey-Bass.

Minocha, S., & Roberts, D. (2008). Laying the groundwork for socialisation and knowledge construction within 3D virtual worlds. *ALT-J, 16*(3), 181–196.

Norman, D. A. (1999). Affordances, conventions, and design. *Interactions, 6*(3), 38–43.

Paliokas, I., Arapidis, C., & Mpimpitsos, M. (2011). PlayLOGO 3D: a 3D interactive video game for early programming education: let LOGO be a game. In *Proceedings of games and virtual worlds for serious applications (VS-GAMES), 2011 third international* (pp. 24–31), 4–6 May 2011.

Papert, S. (1972) On making a theorem for a child. In *Proceedings of the ACM annual conference* (pp. 345–349), August 1972, Boston, Massachusetts, United States.

Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. New York: Basic Books.

Papert, S. (1980s). Constructionism vs. instructionism. Retrieved 10.01.08, from. http://papert.org/articles/const_inst/const_inst1.html.

Papert, S. (1991). Situating constructionism. In I. Harel, & S. Papert (Eds.), *Constructionism* (pp. 1–14). Hillsdale, NJ: Lawrence Erlbaum Associate.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastwood, E., Brennan, K., et al. (2009). Scratch: programming for all. *Communications of the ACM, 52*(11), 60–67.

Robertson, J., & Kipar, N. (2010). Learning together and alone in virtual worlds. In K. Sheehy, R. Ferguson, & G. Clough (Eds.), *Virtual worlds: Controversies at the frontier of education* (pp. 67–84). New York: Nova Science Publishers.

Rosenbaum, E. (2008). Scratch for second life. In S. Veeragoudar Harrell (Chair & Organizer), *Virtually there: Emerging designs for STEM teaching and learning in immersive online 3D microworlds. Symposium in proceedings of the international conference on learning sciences – ICLS 2008*. Utrecht, The Netherlands: ICLS. Abstract retrieved 01.02.10, from http://www.fi.uu.ul/en/icls2008/144/paper144.pdf.

Salmon, G. (2009). The future for (second) life and learning. *British Journal of Educational Technology, 40*(3), 526–538.

Sanchez, J. (2007). Second life: an interactive qualitative analysis. In C. Crawford, et al. (Eds.), *Proceedings of Society for Information Technology and Teacher Education international conference 2007* (pp. 1240–1243). Chesapeake, VA: AACE.

Sanchez, J. (2009). Barriers to student learning in second life. *Library Technology Reports, 45*(2), 29–34.

Savin-Baden, M. (2008). From cognitive capability to social reform? Shifting perceptions of learning in immersive virtual worlds. *ALT-J, 16*(3), 151–161.

Stake, R. E. (1995). *The art of case study research*. London: Routledge.

Tuckman, B. W. (1994). *Conducting educational research*. London: Harcourt Brace College Publishers.

Warburton, S. (2009). Second life in higher education: assessing the potential for and the barriers to deploying virtual worlds in learning and teaching. *British Journal of Educational Technology, 40*(3), 414–426.

Yeh, A. (2010). Three primary school students' cognition about 3D rotation in a virtual reality learning environment. In L. Sparrow, B. Kissane, & C. Hurst (Eds.), *Shaping the future of mathematics education: Proceedings of the 33rd annual conference of the Mathematics Education Research Group of Australasia*. 3–7 July 2010, Fremantle, Western Australia. MERGA.

Yeh, A. J., & Nason, R. A. (2004). VRMath: a 3D microworld for learning 3D geometry. In *Proceedings of world conference on educational multimedia, hypermedia & telecommunications*, 21–26, June 2004, Lugano, Switzerland.

Yin, R. K. (2009). *Case study research: Design and methods*. Thousand Oaks, CA: Sage.