TALES: An E-Learning Application to teach programming concepts to the Early Years Foundation Stage

Melenie Schatynski
Department of Informatics
University of Sussex
ms660@sussex.ac.uk

Sharon Wood
Department of Informatics
University of Sussex
s.wood@sussex.ac.uk

Abstract

As a pupil concludes Key Stage 1[1] they will have been taught to understand basic algorithms, typically in a recipe style, create and debug simple programs and understand why their programs behave the way they do, in addition, gaining knowledge of common technologies. Introducing these concepts to children in their pre-school education will prepare them for the new changes to the curriculum. This study describes the development of an educational system called TALES which is designed to teach programming concepts through a series of mini games. It is aimed at children at the Early Years Foundation Stage (EYFS), specifically children with a cognitive ability between three and six years of age, and therefore does not assume the user to be a fluent reader or writer. The application is evaluated through a small user evaluation study. Early results indicate that learners engage enthusiastically with the games, taking more time to follow instructions and greater care as they become increasingly motivated to succeed in advancing through progressively more challenging levels of the game.

1. Introduction

As technology evolves, the demand for developers and engineers has increased. However, in 2014 only 4% of graduates studied Computer Science at degree level (Higher Education Statistic Agency, 2014). Since then the Department of Education has introduced Computing as a compulsory subject (Department for Education, 2013) in the national curriculum to prepare children for their future in the Digital Revolution. In keeping with these recent curriculum changes, this paper discusses the development of an application designed to educate and interest younger users in a STEM-based subject.

The new curriculum focuses on programming and aims to ensure that children leave education as computational thinkers (Department for Education, 2013). The skills learnt in a computing lesson are strongly linked to other compulsory subjects such as maths, science and design and, although not every school leaver will become a software engineer, the skills acquired will benefit them in other careers and in their personal progression (Resnick, 2012). Programming and debugging code enables pupils to solve problems by logical thinking, whilst constructing their own programs enables them to express their creativity and improve their design skills (Computing at Schools, 2015a). It gives children from all backgrounds a chance to express themselves (Resnick, 2012), share their ideas and have access to technologies they may not have access to at home.

Organisations such as Computing at School (CAS) (Computing At School, 2015b) and BCS (BCS, The Chartered Institute for IT, 2015) have provided teachers with the confidence to teach computing by supplying them with a range of resources (Computing at School, 2015c). QuickStart, established by CAS and Microsoft (Microsoft, 2015), is an online resource aimed at primary school teachers and provides a collection of games and visual programming[2] applications such as Scratch and Lightbot to assist in lesson planning (figure 1). These applications are aimed at different ages and teach children the skills required according to the Department of Education (Computing At School, 2015b).

[1] Education institutes in England and Wales refer to pupils in the first and second year of their education as Key Stage 1, during this stage, pupils are aged between 5 and 7.

[2] A visual programming languages allows a user to create a program through a graphical interface instead of the traditional characters and tokens. In this example the user is able to build their program by arranging "blocks" on the screen.

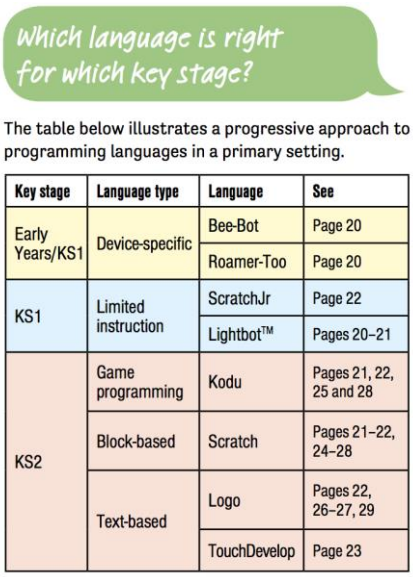

Figure 1 - Extract from Quick Start Computing Guide for Primary School Teachers. (CAS, 2015d, p20)

Although computing is now a compulsory subject in the national curriculum, Academies are not required to follow the same curriculum as state schools (Department for Education, 2015) and an increasing number, 55% in 2015, deviate from the national curriculum (Gee, Worth, & Sims, 2015) which therefore affects the number of children being taught computing in the classroom. Similar to other educational programming languages aimed at the early years (such as Scratch Jr (Lifelong Kindergarten Group, 2016)), TALES is an iPad application that teaches its users about programming concepts. Scratch Jr allows free play, whereas TALES uses a series of mini-games to introduce concepts gradually, using colourful, friendly and entertaining visual representation and graphics and aims, to grasp a child's interest regardless of the subjects they are taught at school.

2. Background

2.1. Education and Development

Educators in the early learning stages use a common framework which encourages children to learn through playing, exploring, active learning and creative thinking (Foundation Years Organisation, 2015). During the EYFS, a child will develop quickly, therefore between 30 and 60 months a child will be assessed against two overlapping developmental stages in accordance with the UK Government and Foundation Years Organisation framework (Department for Education, 2015).

Children between these ages should be able to understand and respond to simple instructions to place objects (including those that involve prepositions) and will begin to understand questions in the form of interrogatives ('why' and 'how'). TALES communicates with the child in the same way by asking questions regarding the characters on the screen, e.g. 'How many steps will Jack need to take to be next to the table". In addition, children between 30-50 months will begin to expand their vocabulary by using conjunctions to form more complex sentences to express their ideas. This enables the application to explain the logic surrounding an action, e.g. 'Jack moves 10 steps because…'. It uses very simple language that enables the user to understand the connection between their choices and what is appearing on the screen.

To explain these ideas, the application provides the user with simple sentences in the form of text on the screen. According to the Foundation Years Organisation children older than 40 months will begin to sound out words and read simple sentences, whereas children with a cognitive age of 30 and 40 months will still be developing these skills, therefore, the application relies on also providing the child with audio feedback and visual explanations to ensure the entire target audience understands.

The decision to create an education application for iPad was due to the rise in assistive technology in classrooms and the many benefits that have been discovered. Recent case studies have shown an

improvement in a child's development after introducing iPads as an educational tool in various K-12[3] schools across the United States. Educational institutions have identified a correlation between iPad assisted learning and their pupils' overall engagement and motivation to learn (Apple Education, 2012). Similarly, in the UK, tablets are now being used in 68% of primary schools providing assistance in everyday lessons (Coughlan, 2014). An additional group to see the benefits of using tablets in the classroom is children with special educational needs. They are being used as an assistive technology to help pupils of special education institutions overcome the additional struggles they are faced with in communication and learning (McKnight, 2013). The addition of tablets has been proven to enable disabled users to become more independent and focused when learning (Queensland Government, 2012).

2.2. Learning with Multiple Representations

The application uses multiple representations to explain challenging programming concepts through animations, text, and audio. The use of multiple representations, in this case, will be to construct a deeper understanding of an idea. Multiple representations have many advantages when presented to the user in a way which is most appropriate to them. In order to grasp the idea successfully, the user must have prior knowledge of at least one of the representations especially when used to assist the interpretation of another (Van der Meij & de Jong, 2003). By keeping the language simple through audio and text representations, they can be used to present complex ideas to younger children and complement what the other is attempting to portray (Ainsworth, 2008).

A popular representation to present abstract complex ideas is through animations. It has been argued (Tversky et al., 2002), that the use of animation adds little benefit to a student's learning. Based on their meta-analysis of existing studies, students performed better because the animations presented provided more information than their static equivalents, therefore, deemed incomparable. However, each study discussed concluded the group of students using the animation to aid their learning exceeded the performance of those using the static graphics. Therefore, it could be argued animations allow for a more detailed description and enable interactive learning. Animations within the application are used to illustrate to the user how the set of instructions interact with each other. In TALES, animations appear slow and clear to ensure the user can interpret each individual instruction in the program sequence.

3. The TALES Application

The objective of this study was to design and implement an application to introduce programming to children in pre-school education as shown in figure 2.

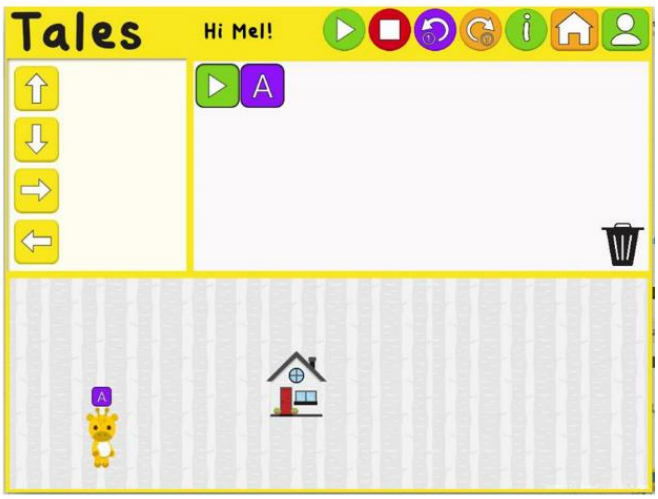

Figure 2 - TALES screenshot

[3] K-12 education refers to a child's education from kindergarten to 12th grade, this is typically children between 5 and 18 years old.

The iPad application (shown in figure 2) prompts a user to construct a program using a visual programming language (VPL) to complete levels. The user receives rewards and unlocks subsequent levels on successfully creating a program sequence that completes a task. The VPL is represented using colourful and engaging instruction blocks. Each instruction block belongs to one of four types and these types are used to validate a user's input once a sequence has been executed. The four types are as shown in table 1. The language has been designed so that a Play object must appear at the beginning of a sequence and Actions appear after Object blocks as an Action belongs to an Object.

Instruction Block Type	Description
Play Block	The Play will appear at the beginning of every sequence to represent the start. It should only appear once in a sequence. The play block appears as a green square block with a white arrow head.
Object Block	An Object block represents a character. A user can animate an Object by placing Action blocks subsequently to an Object block. Objects are labelled using letters of the alphabet.
Action Block	An Action block determines how the Object (character) will be animated. Figure 5 shows yellow Action blocks which allow the character to move, up, down and right.
Control Block	Control Blocks control how each section of the sequence is executed. Two examples of Control blocks are conditionals and loops.

Table 1 - Instruction Block Types

3.1 Level Structure

The app takes the user through a series of progressively more challenging levels of the game. Each level presents the user with new knowledge to absorb (as shown in Table 2), following which the user is asked to complete a short task that applies this knowledge. The user is able to execute their program (figure 3) which plays a short animation that corresponds to their program sequence, providing the user with immediate visual feedback. Small, simple pieces of information (figure 4) are presented to the user to ensure each concept is understood. In addition, concept-relevant feedback is provided when a user incorrectly runs the program to reinforce their understanding of the original concept before they re-attempt the level. The type of feedback presented to the user is dependent on whether a parse or a code error has occurred (figure 5).

Level	Description
1	Level 1 introduces the Action Blocks. A Play and an Object block are presented to the user as an existing and un-editable part of the sequence. The user has the choice of four Action Blocks to append to the program sequence (Up, Down, Right and Left).
2	Level 2 introduces one Object Block and informs the user that Action Blocks change the position of an Object (character). The program sequence still presents the user with an un-editable Play Block, however a user can now place an Object Block anywhere in the sequence in addition to the Action Blocks.
3	Level 3 introduces the Play Block. The user is no longer given any pre-set blocks in their program sequence.
4	Level 4 introduces two Object Blocks. Users are taught that Actions are only applied to the Object Block they appear subsequently to.

Table 2 – Level Progression

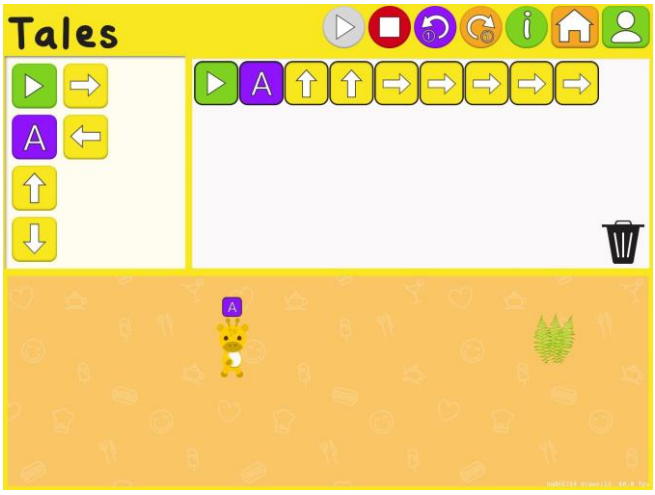

Figure 3 - TALES Level 3

The user receives star rewards once they have completed a level. The user will always receive at least one star and the number of stars is determined by the number of attempts taken to complete each level (fewer attempts leads to more stars). Rewarding users in relation to their number of attempts introduces a gamification element to the application therefore rendering it more engaging.

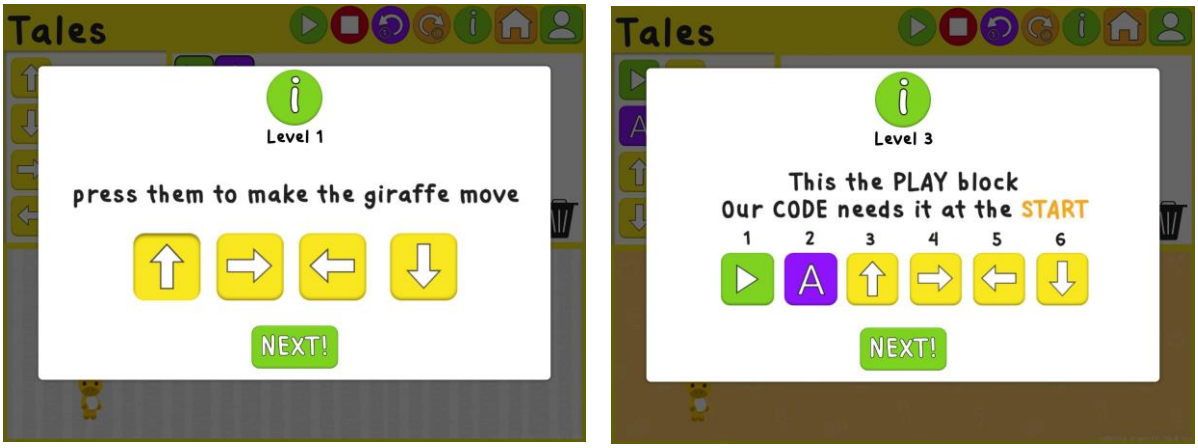

Figure 4 – TALES level progression

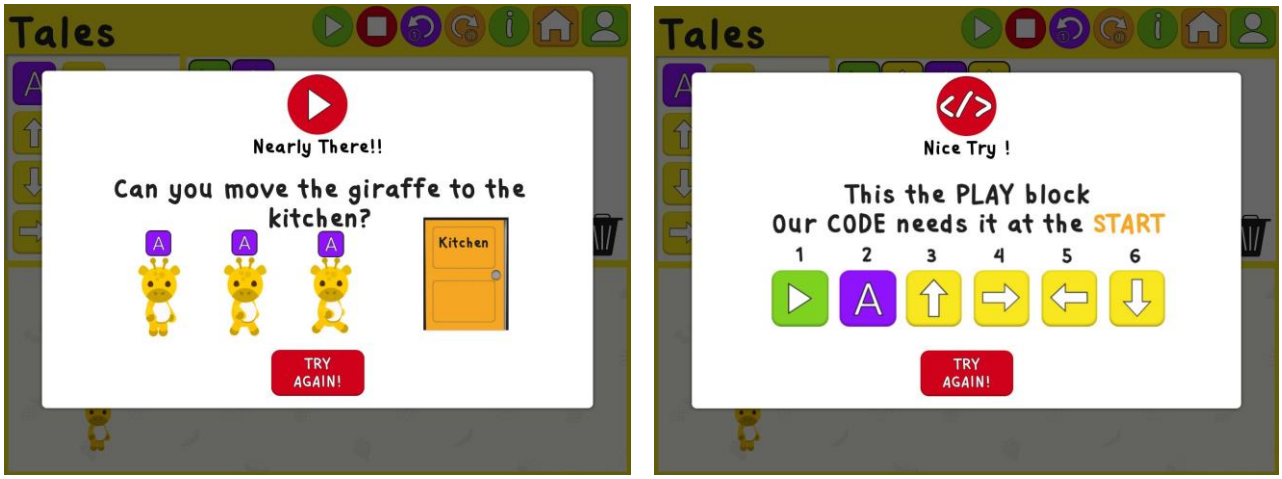

Figure 5 - 3 L-R Code Error Feedback, Parse Error Feedback

To include additional blocks in a program sequence the user can select instruction blocks (top left section in figure 3) to appear in the program flow window (top right section in figure 3). The user is able to alter this sequence by dragging items in the program flow window to the bin to remove completely or by dragging and dropping blocks to amend the program sequence order.

In addition to background research, the app design was additionally informed by a small number of children from the target age group who were invited to suggest ideas for the various app levels and, as a consequence, the design of several levels were based on those suggestions. Figure 6 illustrates one of the suggested levels (to create a cake) used within the game.

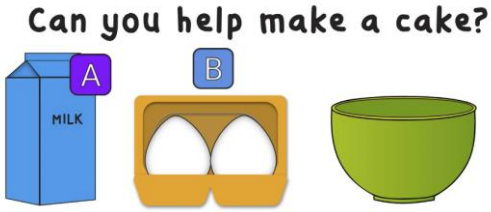

Figure 6 - Instructions from Level 4

3.2 Main Colour Scheme

The choice of colour scheme was a crucial decision in the design process as colours can be associated with a person's behaviour (Doherty, 2010). As the game is designed for young children, it needed to be engaging and fun; therefore, a warm colour was chosen as these are generally associated with stimulation and happiness. In addition, research (Doherty, 2010) shows that users' cognitive performance increases when exposed to warmer colours.

The colour yellow was chosen as the application's primary colour over other warm colours such as red and orange, as red is more commonly associated with danger (Saif, 2011) and recent studies (Saif, 2011) showed that yellow has a more positive connotation than orange. Characters are animated when a user executes their program and have been designed to be colourful and 'cute' to allow for a positive learning environment.

Figure 7 - Main Character in TALES

4. User Evaluation Study

4.1. Method

A group of children between the ages of four and six years old were invited to participate in an observational study in which the children used the app, followed by a short group discussion which allowed each participant to express their opinion regarding the application. Informed consent was obtained from each participant's guardian before the study began. In addition, the purpose and structure of the study along with their right to withdraw was explained to each child and their verbal assent acquired. The focus group was conducted in a home environment to ensure naturalistic results. Participants were recruited based on their age and were of mixed ability. To ensure the data collected is kept confidential each user was assigned a unique number which they were able to use to log into the system. Only the participant's progress data and their unique number is saved, therefore leaving each user anonymous. The participants will be referred to as shown in table 3.

	Age	Gender
Participant A	6 years old	Male
Participant B	6 years old	Female
Participant C	5 years old	Female
Participant D	4 years old	Male

Table 3 - Participants.

Quantitative data about each user's progress was captured automatically by the app and the data was sent to an external source. Each child was provided with an iPad with the application pre-installed and assigned a participant number to use to log in for the purposes of the study. In addition to the automatically captured quantitative data, hand-written observational notes about each participant's interaction with the application were taken throughout the session of playing using the app.

4.2. Data Collection

For the purposes of the user study, TALES records a user's progress to an external database including correct and incorrect program sequences, time to completion and completed levels. If an internet connection is unavailable, the data is stored locally until an internet connection is available. The incorrect program sequences focused on two types of errors, parse and code and were recorded/coded retrospectively by the observer following the procedure shown in figure 8. A parse error occurs when a sequence doesn't conform to the rules of the VPL, whereas a code error occurs when the sequence of instruction blocks is valid but the sequence does not complete the level (e.g. the character will not reach the food as shown in figure 3).

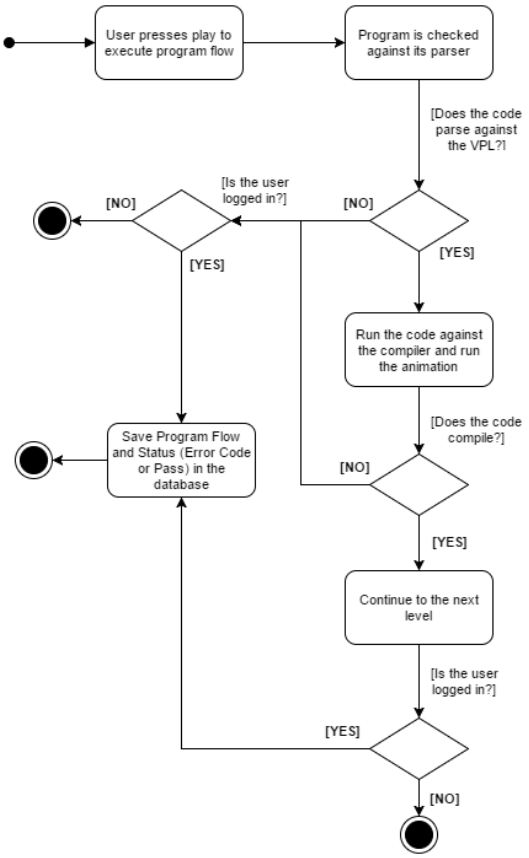

Figure 8 - Validation Process

4.3. Quantitative Results

Each time a user attempted or completed a level, the program flow sequence was stored in an external database complete with any relevant error types.

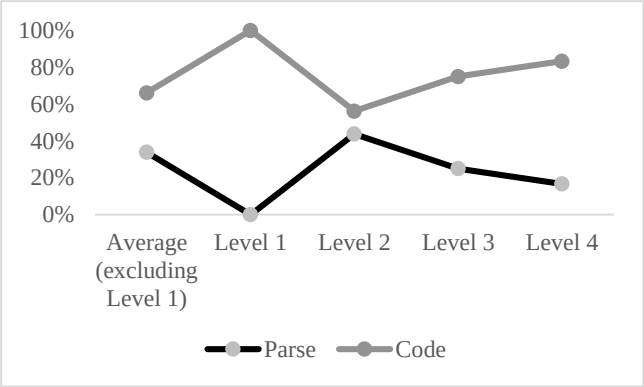

Figure 9 - Percentage of parse and code errors

The graph in figure 9 shows the percentage of each error type per level. As parse errors are not possible in level one they have been excluded from the average.

The graphs in figures 10 and 11 display each user's time to complete and the number of attempts per level, respectively. A comparatively drammatic increase can be seen in the figures relating to level 2 with an equal percentage of code errors to parse errors. The introductory level was completed in the least amount of time with the least amount of attempts by the majority of participants.

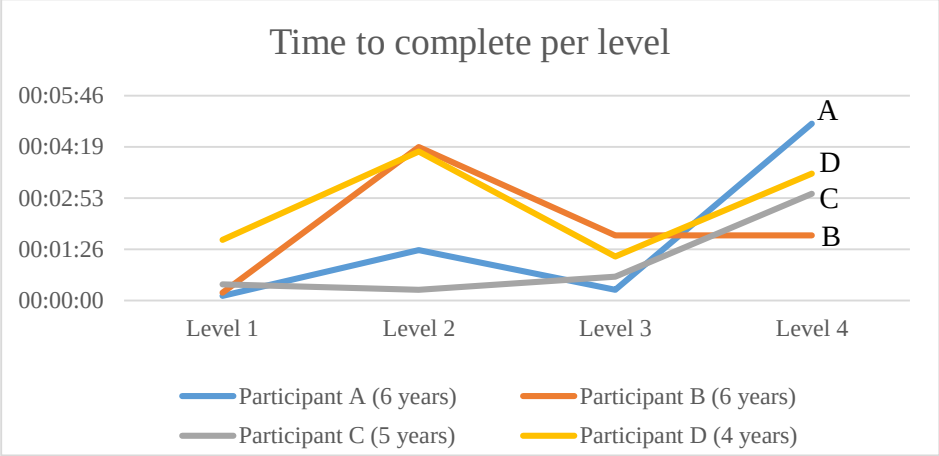

Figure 10 - Graph represents the time taken to complete per level

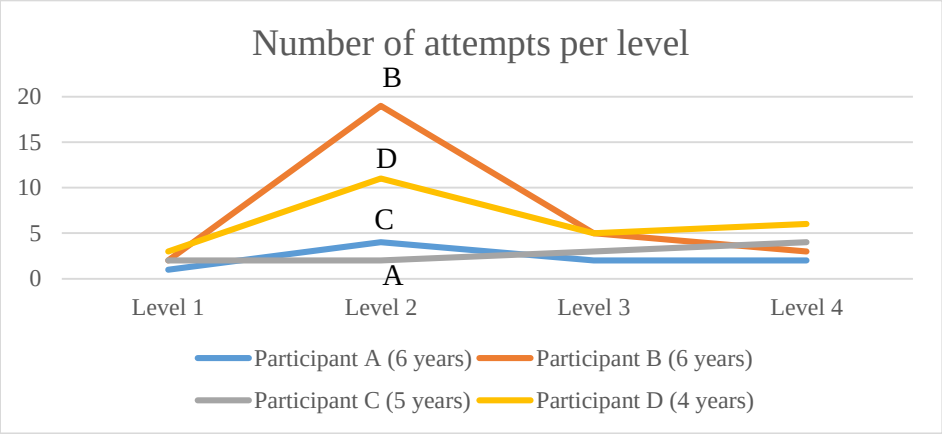

Figure 11 - Graph represents the number of attempts per level

The program flow for each parse error has been investigated and the main issues users faced have been detailed in the table below.

Level the error occurred	Reason for the parse error
Level 2	Participant placed two Object blocks in the program flow without continuing the sequence with Action blocks.
Level 2	Participant placed Action blocks before the Object block.
Level 2	Participant appended an additional Object block to the end of a correct sequence.
Level 3	Participant didn't include a Play block at the beginning of the sequence.
Level 3	Participant didn't include a Play block or an Object block at the beginning of the sequence.
Level 4	Participant included both Object blocks correctly, although didn't include the Play block at the beginning of the sequence
Level 4	Participant attempted to append an Action block to two Object blocks, each Object block requires its own Actions.

Table 4 - Program Flow Errors

4.4. Observation Results

When the children were presented with the application they seemed to be excited to get started. Therefore, a couple of the children started the levels before logging into the system. This did not pose a problem as the system logs a user's progress locally and uploads it once they have logged in. All participants seemed to understand the first level without any problems, although there was a bit of trial and error to find out how far the giraffe would walk based on a single Action block.

Participants seemed to enjoy the star rewards and felt a sense of achievement when they were received and were happy when they progressed to a new level. Participant A regularly expressed their joy when they received three stars for completing a level.

Participant B would remove blocks rather than rearrange if there was an error in their sequence, as they appeared to be unaware of the drag and rearrange feature in the program sequence.

The second level seemed more difficult for the majority of users. Users were eager to progress past the instructions to the actual game, and therefore didn't read the instructions. In addition Participant D (the youngest participant) was confused about which button needed to be pressed in the Feedback view. Participant D also made the most parse errors and often expressed frustration with the application.

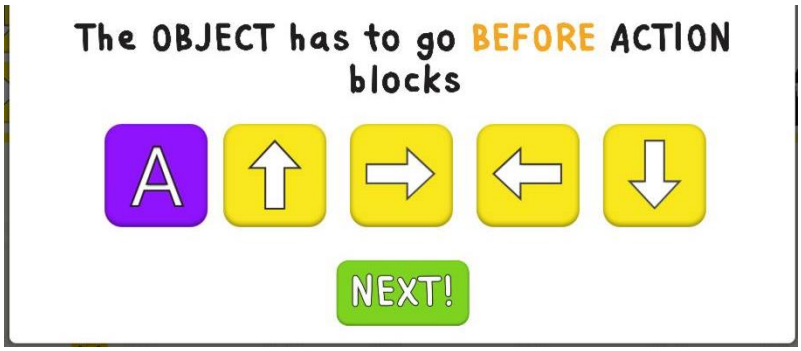

Figure 12 - Feedback View from Level 2

The majority of participants became more focused whilst completing later levels, appearing to spend more time thinking about the solution before attempting it, and would express their delight on completing another level.

All the users touched the *Play block* in the program at least once to try and execute the program instead of the one in the menu bar and none of the users used the undo or redo functions. The older participants in the group didn't require any assistance, whereas the youngest participant only started to enjoy the game once the aim of each level was read to them verbally. However, the language used within the application was understood by all participants.

From the discussion it was discovered the participants felt like they learnt something new. When asked what they liked most about the application, several participants answered level 3 as they felt that was the easiest level. Participant B found the application 'nice' and 'challenging' although at times confusing, in particular the number of steps needed to get to the end. Participant D didn't feel as though they would play the game again as they found it too difficult, whereas Participant C enjoyed playing the game and would play again. All participants expressed the view that their favourite level was Level 4 as they found it challenging and fun.

5. Discussion

This and the following sections discuss the results presented above and evaluate the application in terms of the primary objectives of the study, suggesting improvements based on those findings.

5.1. Does the application teach the user basic programming concepts?

The application presents the user with new knowledge and requires them to complete a task based on that knowledge. Whilst observing users with the application, it was noted that they made frequent mistakes regarding the program sequence which were typically code errors. The discussion that followed and the data logged by the system revealed this was due to the user's inability to gauge the distance attached to each Action block throughout all levels. Therefore, the vast number of code errors are due to the participants' trial and error approach to the program sequence. Taking this into account, we can start to look at the parse errors[5] that occurred which, as shown in figure 9, are significantly fewer. Only 33% of the errors that occurred were due to a parse error and, of those, 73% were generated by one participant. This suggests the remaining participants understood the rules of the VPL.

The results show, a decrease in the number of attempts for all participants in the final two levels (figure 10), however, the time to complete increased in the final level. This could suggest that participants were taking more time to solve the problem in later levels as they began to understand the concepts more. Although parse errors did appear in later levels, the number of parse errors steadily declined (figure 9) and the observations indicated that participants seem to learn from their earlier mistakes and rectify their mistakes quicker than for previous levels, despite the level of difficulty increasing.

The initial introduction of Object blocks in Level 2 confused several participants as more errors occurred at this level (shown in figures 10 and 11). As discussed in the observation study results section, participants were eager to progress and skipped the instructions, therefore, allowing insufficient time to absorb the new knowledge. A possible solution to this is discussed in the next section. Users did not disregard the level introduction in later levels following Level 2. In addition, after Level 2 the amount of time to complete increased (figure 10) as the number of attempts (figure 11) stayed consistently low: to ensure the information was understood and they could complete the level with fewer mistakes, users took greater care when reviewing the instructions at the start of each level.

Overall participants seemed happy with the application and seemed confident in using it. The majority of participants expressed they had learned something new and found the game enjoyable. Although the positive results are indicative, it is important to confirm them by performing a user study involving a larger sample size in order to conclusively establish that the application consistently teaches programming concepts to its users.

[5] Parse errors are generated when the program flow sequence will not parse in the current order.

5.2. Suggested improvements based on findings

Several issues were raised during user testing, outlined below with suggestions for potential solutions.

As discussed, users were unsure about the distance each Object would move in relation to the Action blocks they chose, leading to frequent code errors. This problem could be resolved easily with the addition of a grid being placed on top of the existing background as a guide, enabling the user to count the number of cells the giraffe would need to move to reach the desired position.

Younger users were less inclined to read the instructions at the beginning of each level. The addition of audio instructions could aid the user in their learning experience, removing the need for additional assistance. To ensure confident users weren't hindered by the addition of audio instructions, this would be a feature the user could activate and deactivate as and when they required. Another solution to the tendency to skip instructions could be to present the user with a short quiz, testing them on the new information presented, before allowing them to continue.

A number of the participants struggled to differentiate between buttons and images in the feedback pop up. A simple solution to this would be to redesign the images without the raised effect that misleadingly suggests they are actionable buttons.

Participants attempted to play the animation using the Play block in the *Program Flow*. This could be resolved by using separate icons for the Play block and the Play button, or alternatively allowing the Play block to also validate the program as many participants automatically chose the Play block before the Play button.

Some of the participants had a much higher number of attempts on level 2. This could be resolved by offering the user a hint about the next possible block. The game was designed using a 2D array to represent a grid, therefore, this could be achieved using a search algorithm to find a pathway to the End from the Object's current position and suggest the next block to the user.

6. Future Work

In addition to the improvements suggested in the previous section, an obvious improvement would be to extend the range of levels within the game to include further programming concepts, such as conditional and loop statements (Control blocks), thus increasing the user's knowledge of basic programming concepts. A further idea is to incorporate a Project Mode within the game. The aim of this would be to enable users to create their own additional game levels and animations using the skills they have acquired from the main game.

Further investigation through user testing would support the suggested improvements such as experimenting with different versions of the Play block and Play button. In addition, to ensure educators are comfortable using TALES as an educational tool in their classrooms, further user studies are required using primary and early year's teaching staff.

7. Conclusion

This paper presents an app developed to assist young children to grasp programming concepts early in their educational experience. Overall, the application was a success and through user testing it was found to increase interest in STEM subjects (notably programming) amongst the young users involved.

Since this project began, educational technology aimed towards computing continues to evolve. The introduction of the Raspberry Pi Zero (Raspberry Pi, 2015) in late 2015, continues to make computer science more accessible and affordable to the masses, rendering it all the more important to equip young people with the concepts needed to exploit these technologies and to forge productive careers.

It is intended to continue the development and research to improve TALES considerably in the future. Ultimately, it is hoped to release it as a fully functioning application available through Apple's App Store.

8. Acknowledgements

The authors would like to thank Marianna Obrist and the members of her research group for the equipment used in this study, the parents and children who took part in testing the application, whose

contributions were crucial to developing the TALES application, and Judith Good for her continued encouragement in the preparation of this paper.

9. References

Ainsworth, S. (2008). The educational value of Multiple-Representations when Learning Complex Scientific Concepts. . In Gilbert JK.; Reiner, M; Nakhleh,M (Eds.) Visualization: Theory and Practice in Science Education.

Apple Education. (2012). *iPad in Education Results.* March, 2012. Retrieved October 2015, from https://www.apple.com/education/docs/iPad_in_Education_Results.pdf

BCS, The Chartered Institute for IT. (2015). *BCS, The Chartered Institute for IT*. Retrieved September 2015, from http://www.bcs.org/

CAS. (2015d, p20). *QuickStart Computing. A CPD toolkit for primary teachers.* Retrieved from http://primary.quickstartcomputing.org/resources/pdf/qs_handbook.pdf

Computing At School. (2015b). *Computing At School. Educate, Engage, Encourage*. Retrieved September 2015, from http://www.computingatschool.org.uk/

Computing at School. (2015c). *CAS Barefoot Example Activities*. Retrieved October 2015, from http://barefootcas.org.uk/barefoot-primary-computing-resources/exemplar-activities/

Computing at Schools. (2015a). *Computing in the national curriculum. A guide for primary teachers.* Retrieved November 2015, from http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

Coughlan, S. (2014). *Tablet computers in '70% of schools'.* December, 2014. Retrieved October 2015, from http://www.bbc.co.uk/news/education-30216408

Department for Education. (2013). *Statutory Guidance. National Curriculum in England: computing programmes of study*. September, 2013. (British Government) Retrieved November 2015, from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

Department for Education. (2013). *Statutory Guidance. National Curriculum in England; computing programmes of study*. Retrieved from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

Department for Education. (2015). *Policy Paper. 2010 to 2015 government policy: academies and free schools.* Retrieved October 2015, from https://www.gov.uk/government/publications/2010-to-2015-government-policy-academies-and-free-schools/2010-to-2015-government-policy-academies-and-free-schools

Department for Education. (2015). *Statutory framework for the early years foundation stage.* Retrieved September 2015, from https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/335504/EYFS_framework_from_1_September_2014__with_clarification_note.pdf

Doherty, A. (2010). Effects of Environmental Colour on Mood: A wearable Life Colour Capture Device. In *Proceedings of the 18th ACM international conference on Multimedia* (pp. 1655-1658). ACM.

Foundation Years Organisation. (2015). *Greate early years & childcare.* 4Children. Retrieved from http://www.foundationyears.org.uk

Gee, G., Worth, J., & Sims, D. (2015, May). *Academies and maintained schools: what do we know?* (Full Fact) Retrieved October 2015, from https://fullfact.org/education/how_good_are_academies_compare_maintained_schools-42769

Higher Education Statistic Agency. (2014). *Online Statistics - Students by subject area, level, mode and sex*. Retrieved November 2015, from https://www.hesa.ac.uk/stats

Lifelong Kindergarten Group. (2015). *Scratch*. Retrieved October 2015, from https://scratch.mit.edu/

Lifelong Kindergarten Group. (2016). *Scratch Jr*. Retrieved from https://www.scratchjr.org/

Lightbot TM. (2015). *Lightbot. Solve Puzzles Using Programming Logic*. Retrieved October 2015, from http://lightbot.com/

McKnight, L. (2013). Apps that make things, not apps that do things": appropriation and assistive learning technologies. *BCS-HCI '13 Proceedings of the 27th International BCS Human Computer Interaction Conference*.

Microsoft. (2015). *QuickStart: A CPD toolkit for primary teachers*. Retrieved October 2015, from http://primary.quickstartcomputing.org/

Microsoft, & Computing At School. (2015). *Programming. Introduction to Programming*. Retrieved October 2015, from http://primary.quickstartcomputing.org/resources/pdf/programming.pdf

Preece, J. (2002). *Interaction Design: Beyond Human-Computer Interaction.* John Wiley & Sons, Inc.

Queensland Government, E. (2012). *iPads in Special Education trial report.* Retrieved October 2015, from http://education.qld.gov.au/smartclassrooms/documents/enterprise-platform/pdf/ipad-special-education-report.pdf

Raspberry Pi. (2015). *Raspberry Pi Zero: The $5 Computer*. Retrieved April 2016, from https://www.raspberrypi.org/blog/raspberry-pi-zero/

Resnick, M. (2012). Let's teach kids to code . (T. Talks, Interviewer) Retrieved from https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code?language=en

Saif, M. (2011). Colourful language: measuring word-colour associations. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics* (pp. 97-106). Association for Computational Linguistics.

Sean Coughlan, B. (2014). *Tablet computers in '70% of schools'*. Retrieved October 2015, from http://www.bbc.co.uk/news/education-30216408

Smith, M. J., & Salvendy, G. (2007). *Human Interface and the Management of Information. Methods.* Beijing, China.

Tech City UK. (2015). *Tech Nation. Powering the Digital Economy.* Retrieved October 2015, from http://www.techcityuk.com/wp-content/uploads/2015/02/Tech%20Nation%202015.pdf

Tversky, B., Morrison, J., & Betrancourt, M. (2002). Animation: can it facilitate? *In. J. Human-Computer Studies 557*, 247–262.

Van der Meij, J., & de Jong, T. (2003). Learning with Multiple Representation. *Proceedings of EARLI Conference, Padua, Italy, August 2003.*