Learning Logo at a high school: Constructionism versus Objectivism

Stéphane Norte, *snorte@ualg.pt*
UAlg Informatics Lab, DEEI-FCT, University of Algarve, Campus de Gambelas 8000-117 Faro, Portugal.

Paulo A. Condado, *pcondado@ualg.pt*
UAlg Informatics Lab, DEEI-FCT, University of Algarve, Campus de Gambelas 8000-117 Faro, Portugal.

Fernando G. Lobo, *flobo@ualg.pt*
UAlg Informatics Lab, DEEI-FCT, University of Algarve, Campus de Gambelas 8000-117 Faro, Portugal.

Abstract

In the world of education, the traditional teacher-centric pedagogy is often supported by the traditional lecture format. This methodology however, may not be the best way to assist students to learn in a computer science classroom. We have made different activities in high school to understand students' difficulties to learn the Logo programming language. This paper describes the activities made in two classes which belong to a multimedia course with a specially adapted education syllabus and the results obtained.

The population of the study consists of students from a high school in the south of Portugal who never learned a programming language before, and that have exhibited many learning difficulties and behavioural problems in the past. Some of the students are quite old with respect to their school year. Under normal circumstances, a 10^{th} grade student should be about 15-16 years old. The fact that some of them are much older than that is due precisely to their learning difficulties and behaviour problems, resulting in a large fraction of student failures at the end of the school year. Those students are required to repeat the school year once more. Indeed, a lot of students have failed several times.

Each group of students was exposed to Logo programming using different pedagogical models. We employ the traditional teacher-centric pedagogy (objectivism) in a class and apply the constructionist approach to another class. The study was made throughout 5 months with 180 minutes per week.

It is important that educational systems worldwide progress towards a better future with new generations and new philosophies of education. The research described in this paper supports the notion that the role of the teacher in the classroom needs to change, and that the possibility to learn by exploration must be given to students. Specifically, it encourages prospective teachers to adopt Logo under a constructionist learning environment. The teacher is essential in the learning process to provide motivation and to aid the students in the exploration of new skills.

Keywords

Logo, constructionism, objectivism, learning, education

Introduction

Education and learning in today's computer science classroom is usually based in two philosophical paradigms: objectivism and constructionism (Wulf, 2005).

Objectivism has dominated the field of education for several years. An objectivist educator believes that learning generally occurs when students listen to a teacher's explanation (Fosnot, 1996; Skinner, 1953). This traditional paradigm often manifests itself in teacher-centered and teacher-controlled classrooms. Under this pedagogical approach, the instructor serves as a subject matter expert who is the principal source of knowledge. Learning is seen as an information transfer procedure in which the instructor imparts his or her knowledge to the students. Constructionists, on the other hand, focus more on the experiences of the student. The basic and most fundamental assumption of constructionism is that knowledge does not exist independent of the learner, knowledge is constructed by experimental exploration, by making things (Kafai and Resnick, 1996; Papert, 1993; Papert, 1996).

With the development of new technologies in schools, teachers can follow a less traditional paradigm and employ a more constructionist approach (Gorp and Grissom, 2001). Unfortunately, some schools are not technologically and pedagogically prepared to encourage teachers to use these technologies on a frequent and sustained basis to enhance student learning. The challenge should be about using new approaches to help students to learn concepts and skills embedded in computer science education (Watt, 1988). It is not so much about powerpointing, spreadsheeting or word processing. The focus should be on teaching and learning strategies that make a difference in daily practice, on activities translating into stronger student performance.

This paper describes a research study done at a high school about learning computer programming by different pedagogical paradigms. Our work focuses on Logo because it is an excellent tool for learning introductory programming. Using Logo programming language is also an excellent introduction into the world of computer science education. We decided to employ the traditional teacher-centric pedagogy (objectivism) in a class and apply the constructionist approach to another class.

The paper is organized as follows. The next section describes the population that participated in our study. Then, we present our experience in teaching Logo, both under an objectivist and a constructionist environment. Following that, two sections, one on student feedback and another on student evaluation are presented. The paper ends with a summary and the major conclusions drawn from this work.

Research setting

The population of the study consisted of two classes from the 10th grade of a high school in Algarve, Portugal. Students from both groups never learned a programming language before and are known in the school for having learning difficulties, lack of concentration, and bad behaviour inside the classroom.

The two classes are referred to as the M class and the N class. Both belong to a multimedia course with a specially adapted syllabus. The M class is constituted by 19 students between the ages 16 and 20, with 7 girls and 12 boys. The N class is constituted by 21 students between the ages 16 and 23, with 6 girls and 15 boys. Notice that some of the students are quite old for being still at the 10th grade. Under normal circumstances, a 10th grade student should be about 15-16 years old. The fact that some of them are much older than that is due precisely to their learning difficulties and behaviour problems. The result is that a large fraction of the students fail at the end of the school year and have to repeat it again the following year. Some of the students have failed already several times.

Each group had lessons of Logo programming using different paradigms. In the M class we decided to use a traditional educational approach and in the N class we used the constructionist approach. The study was made throughout 5 months with 180 minutes per week.

Traditional environment

The M class began the introduction to Logo programming by listening to the instructor's explanation and responding to external motivation. We used screen capture video and narrated power point presentations to introduce the Logo turtle motion. The Logo interpreter employed in the beginning of the course was MicroWorlds free demo. We explained to the students how to use the MicroWorlds interface and we spent a lot of lessons describing all features.

After the explanation, each student tried the MicroWorlds environment with the aid of a teacher. Once they have learned the software we began to introduce the Logo vocabulary. With a digital projector we presented some vocabulary and simple procedures to draw geometric shapes. The students drew simple shapes with one defined by us with the turtles using simple Logo vocabulary (see Figure 1).

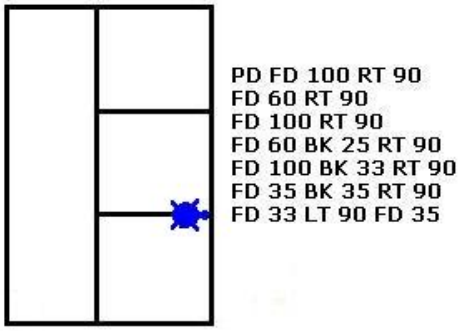


Figure 1 – First shape drawn by the students

When they had defined a few shapes on their own, students were challenged to create some common shapes such as square, triangle, pentagon, hexagon, heptagon and octagon. The students demonstrated a lot of difficulties to create the hexagon, heptagon and octagon. Only two students were able to make the shapes without any help. Figure 2 shows the shapes created by them.

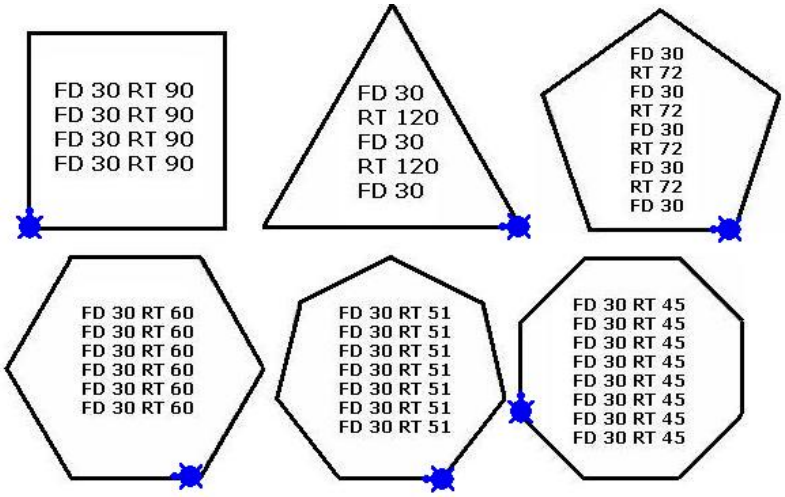
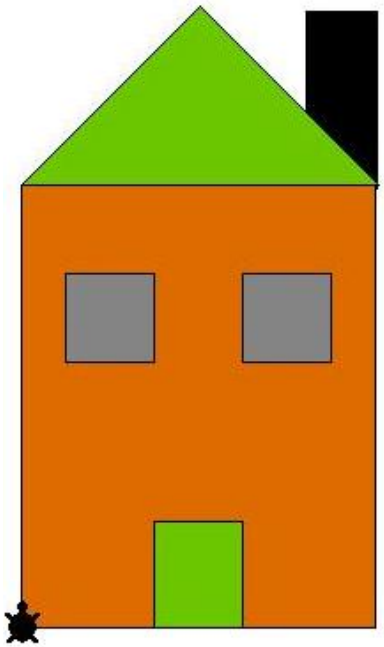


Figure 2 – Shapes created by the students on the M class

Once they had created some shapes, we introduced the basic idea of Logo procedures. This activity allows students to understand some essential geometric ideas and to see the potential of procedures to build shapes. Throughout the lessons, we realized that the activities of the students were only focused on doing precisely what we said and none of them had the curiosity to explore the Logo programming language a little more. The questions were very poor and we detected that the students memorized the Logo commands and procedures instead of understanding the Logo language.
The next step we made was to introduce the “repeat” command. We spent some lessons showing examples of the “repeat” command and asked the students to use it to create shapes. But after the explanation we realized that the students did not understand the functionality of the repeat command. We decided to change the style of the lessons and created more practical activities (Watt, 1988). Throughout the lessons, with a lot of exercises, the students learned the “repeat” concept and we observed improvements in their performance.

The next activity was to explore the creation of a house using procedures, repeat commands and other commands learned in the course. The major problem felt by the students were associated with mathematical concepts. While drawing the house roof, the students had a lot of problems to place the roof in the correct position in accordance with the remaining portion of the house. To help the students with this problem we explained the Pythagoras' theorem. The goal of this activity was to observe students’ skills in building a house with the concepts learned in class. But we realized the students only use simple commands and never the “repeat” command.



```
to House

pd fd 250 rt 90
fd 200 rt 90
fd 250 rt 90
fd 200 rt 180
fd 75 lt 90
fd 50 rt 90
lt 90 fd 10
rt 90 fd 50
rt 90 fd 60
pu rt 180 fd 150
pd fd 50 rt 90
fd 50 rt 90
fd 50 rt 90
fd 50 pu
fd 50 rt 90
fd 50 pd
lt 90 fd 50
lt 90 fd 50
lt 90 fd 50
lt 90 fd 50
pu fd 50
lt 90 fd 75
rt 90 rt 45
pd fd 143
rt 90 fd 143
lt 135 pd
fd 100 lt 90
fd 40 lt 90
fd 55

end
```

Figure 3 – House drawn by the students with Logo

We asked to the students why they had not used the “repeat” command and the answer was “The Teacher did not say that we needed to use the repeat command in this activity”.

The last activity was to analyse syntactically and logically buggy code and write the modified code in MicroWorlds.

Constructionist environment

In our society, schools focus more in traditional pedagogy instead of using different strategies of learning. We decided to develop a course centered in a pedagogy typified by experimental discovery learning through exploration (Papert, 1993; Papert, 1996). The only material given to the students in the classroom was the computers with MicroWorlds. In the beginning of the course the only help we gave was the explanation of the help options inside MicroWorlds. The students were forced to explore the programming concepts on their own. To motivate the students we showed some works made by other pupils and we explained the potential of Logo programming. We realized then that the students manipulated and learned various Logo vocabularies faster than the other class. We were surprised to see the students using advanced vocabulary such as: *setshape*, *touching?*, *forever*, *waituntil*, *repeat*, and so on.

The students learned the Logo turtle motion and the MicroWorlds interface in a fast way. We challenged them to create shapes with the minimum of programming vocabulary. Almost all of them used the "*repeat*" command to draw the shapes. The next activity given was the construction of the house. Some students did not make it because they said that it was very easy and they preferred to create Logo animations. While some of the students made drawings of a house using simple and advanced Logo vocabulary, others created animations with horses, motor-cycles and plants (see Figures 4 and 5).

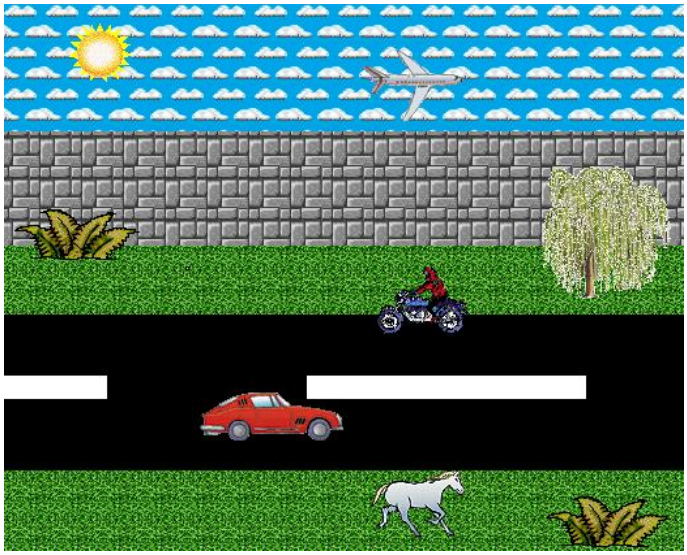


Figure 4 – Drawing made by the students with horse, motor-cycle and plants animations

Figure 5 – The move horse procedure

During a lesson a student asked if it was possible to create a game in Logo. We said of course it is possible. The entire class exchanged a lot of ideas and all the groups said: "We want to create

a labyrinth game with a turtle inside". The idea of the students was to create 3 levels: an easy, a medium, and a hard labyrinth. But inside the labyrinth some rules were created. The turtle can never touch the wall. If that happens the turtle moves to the beginning of the labyrinth. When a turtle finds the end of the labyrinth it automatically changes to another level. To move the turtle inside the labyrinth it is required to use the buttons. When a turtle wins the last levels, a picture appears.

Throughout the course the students worked a lot on this project and all the groups showed a strong motivation. Figure 6, shows two labyrinths created by the students.

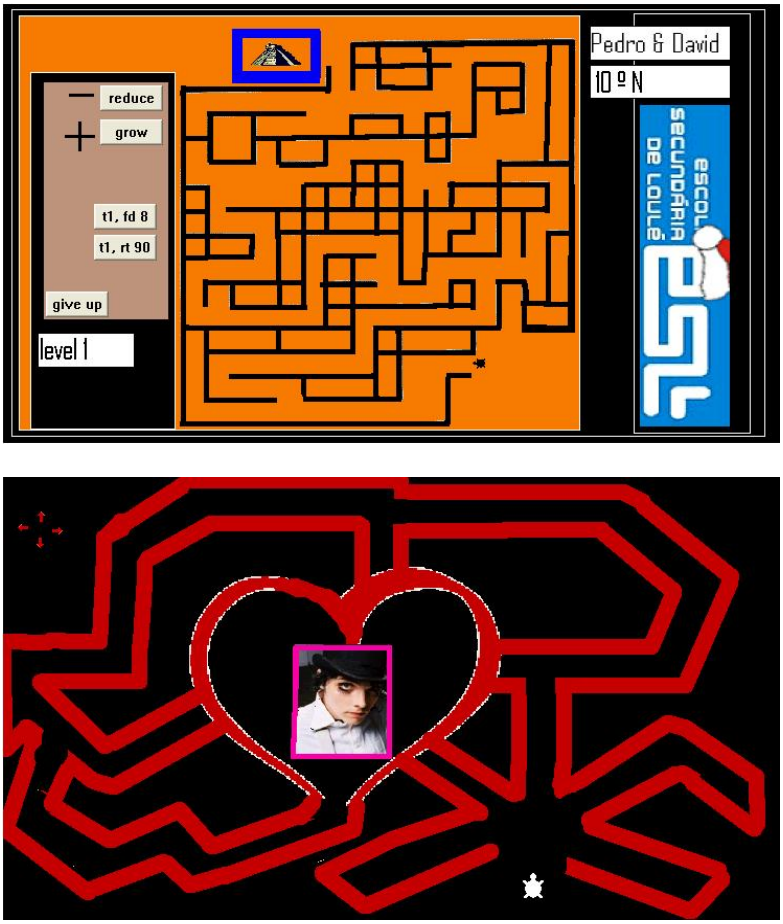


Figure 6 – Labyrinth game made by the students

The students created procedures to build the labyrinth. The most important procedures made were the walls and the transition to another level. Some groups made different procedures (see Figure 7). A group used the "*touching?*" command to change the level and other groups used the "*colorunder*" command. Both groups use the "*colorunder*" command to disallow the turtle to cross the walls.

```
To Walls
Turtle1, forever
[
if colorunder = 15
[setpos[-23 -104]]
]
end
```

```
To ChangeLevels
t1, forever
[if touching? "t1 "End1
[setpos[-340 84]
level2]
]
end
```

```
To Levels
turt1, forever
[if colorunder = 105
[setpos[-340 184]
level2]
]
end
```

Figure 7 – Procedures made by students in the labyrinth game

The goal of the students was to build a hard labyrinth to exchange the game between them and see who would be capable of arriving at the end of the game. After the creation of the labyrinth the students created an HTML template to play the game on the web. The whole work was made by themselves with some suggestions of the teachers.

Student feedback

At the beginning of the course, the students of the N class did not understand the methodology used in the classroom. They asked a lot of questions such as: “But how are we going to learn the Logo programming language alone by ourselves?”, “This is like a big project!”. All the questions had a traditional style.

On the second lesson a positive sign grew in the classroom. The students interacted with each other and constructed a good environment in the class. Many students discussed the motion of the turtle and others talked about the MicroWorlds interface. The motivation of the students grew as the lessons went by, and we can even say that they also became instructors since the environment in the classroom encouraged them to help their colleagues. The students’ feedback to the constructionist environment was very positive and they understood the concepts of learning by doing. We observed a change of mentalities inside the classroom.

The behavior of the M class contrasted sharply with the one observed in the N class. The students made only a few questions and the interaction in the classroom was very limited. Their activities were only focused on doing exactly what we said and they showed little signs of motivation and enthusiasm.

Student evaluation

For both classes, we finished the course with a final exam. Truth be told, we were reluctant to give a final exam to the N class. After all, the idea of a final exam with a set of questions with known answers is something that goes against the spirit of constructionism. Nevertheless, we had to give the exam because it was a formal requirement of the school. A final grade had to be given to each student and, according to the school regulations, it could not be given based on a subjective evaluation.

The exam had three questions limited to 35 minutes and was given to both classes at the same time. The exam dealt with the creation of some shapes using the Logo vocabulary learned in the course, as well as with building simple animations.

During the exam, the students of the M class showed some difficulties in programming the shapes and were worried about the time. The exam produced a stress reaction in them. Their ability to program in Logo was more theoretical and less focused in the discovery process. The students were not stimulated to learn alone and explore the Logo programming vocabulary. In this class we perceived a bad reaction during the exam. When a student did not obtain a positive result the first reaction was to quit. The N class showed a more relaxed behaviour. The students were more calm to accomplish the exam and did not complained.

The final grades describe the perspective we expected. The N class had better grades. The average grade for the N class was 12,4 and for the M class was only 10,1. Note that in Portugal, a 20-point grading scale is used, in which 20 is the highest grade and 0 is the lowest.
We would like to emphasize that, in our opinion, there was no need of doing a final exam in order to conclude that the students of the N class had a better learning experience (and performance). By looking at the activities conducted on both classes, it became evident that the N class was learning much more, was having more fun, and were even doing things that we thought they were not capable of doing. The results of the exam just confirmed what we already felt. A possible explanation for the better performance of the N class during the exam might be due to the enthusiasm towards the subject matter. By the end of the course, the N class really enjoyed programming in Logo while the M class showed no particular interest in it.

Summary and Conclusions

This paper presented two different pedagogical paradigms used in an introductory computer science classroom at a high school in Portugal. The study supports the notion that constructionist activities can aid learning better than a traditional environment.

There is a great interest in the exploration of constructionism in learning environments. In the near future, we intend to create more experiments in high school with students with special educational programs. We also intend to disseminate the Logo programming for other teachers to use in their classrooms.

It is important to try to adopt the constructionist approach in order to open new mentalities in schools. It is important that the educational system progresses for a better future with new generations and new philosophies of education. With this research we believe than the role of the teacher in the classroom needs to change, and the possibility to learn by exploration must be given to students. Any person possesses the ability to learn and discover. The teacher is essential in the learning process to provide motivation and to aid the students in the exploration of new skills.

Acknowledgments

We sincerely thank the contributions of the students who made this study possible. This work was sponsored by the Portuguese Foundation for Science and Technology (FCT/MCES) under grant POCI/CED/62497/2004 and by Foundation Caloust Gulbenkian under grant Proc. 65538.

References

Fosnot, C.T. (Ed.). (1996). *Constructivism*: *Theory, perspectives, and practice*. New York, NY: Teachers College Press.

Gorp, M. J. and Grissom S. (2001) *An Empirical Evaluation of Using Constructive Classroom Activities to Teach Introductory Programming*. Computer Science Education, Vol. 11, No. 3, pages 247-260.

Papert S. (1993), *Mindstorms: Children, computers, and powerful ideas* (2nd ed.), New York: Basic Books.

Papert S. (1996), *The connected family: Bridging the digital generation gap*, Longstreet Press, Inc.

Kafai, Y. and Resnick, M. (eds) (1996): *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World.* Mahwah, New Jersey,Lawrence Erlbaum.

Skinner, B.F. (1953). *Science and human behavior*. New York, NY: Free Press.

Watt, D. H. (1988) *The Computer as Microworld and Microworld Maker.* In SIGCUE Outlook, ACM press, New York, USA, Vol. 20, No. 1, pages 81-89.

Wulf, T. (2005) *Constructivist approaches for Teaching Computer Programming*. In SIGITE '05: Proceedings of the 6th conference on Information Technology Education, ACM press, October, New Jersey, USA, pages 245-248.