Comparison of the Learning Curve and Adaptive Behavior from Kids to Adults who Create Mobile-Apps and Little Robots Using Block-Programming

Felipe Moreno-Vera, Universidad Católica San Pablo, Perú
Leonardo León-Vera, Universidad Nacional de Ingeniería, Perú
Juan Guizado-Vasquez, Universidad Nacional de Ingeniería, Perú
Michael Vera-Panez, Universidad Nacional de Ingeniería, Perú

The Asian Conference on Education 2018
Official Conference Proceedings

Abstract
Block programming presents an interactive and very simple way to learn to program, today block programming applications allow you to develop and program the electronic hardware components such as sensors and motors, whose relationship between hardware, software and mobile applications are fundamental in this technological age. In this article we present a study on how much the speed of learning differs and how much information retention capacity children, adolescents and adults have in the same conditions of learning, environment, tools and teaching system with the topic of creating robots through simulation of electronic circuits. In addition, the manipulation of electronic components such as sensors, motors and Bluetooth is presented.

Keywords: Learning, Education, Robotics, Kids, Teenagers, Adults, Programing languages, Block Programming

iafor
The International Academic Forum
www.iafor.org

Introduction

In the context of Peru, the education methodology is not enough to a complete a certain level of knowledge. Our target study subjects are people between the ages of 9 and 45, because is the 57.232% of the total population (information was collected from [1]). Kids (6-9 years old) have more easily way to learn and play with new tools [2], Teens (10-12 years old) and Juniors (13-17 years old) have the same capacity to learn, but they have another distractions that causes a little reduction in the learning curve for new things [3].

Adults (from 18 years old and on) have a different way to learn things about technology, some of them have experience working with computers but others do not use computers in daily life. We have approximately 100 children, 100 teens, 100 junior and 100 adults, each group is divided into a group of 25 people with the same curriculum, the same materials, the same teachers and the same 16 lessons (each lasts 3 hours a day).

Currently, in Perú we have a lot of devices distributed in all families, adults, teens and kids. Is very common see a kid with a tablet or mobile phone at 8 years old playing games or using social networks (with parental control like facebook) or watching videos about youtubers (gamers with minecraft channels, fornite, etc).

In this context, we have technology in our hands every day any time, but in education until this time, schools separate technology for the classrooms, schools don't use technology to teach and don't tech how to use technology responsibly. So, this create a disorder in the generation gap between people who use technology only for play games and those who do not use technology to improve techniques or methodologies to teach or learn new things.

In other words we have population that explode technology in any another aspect except in education and our consequence is that schools don't teach programming at early ages, our young people learn computer concepts or how to programming in the university at age between 20 and 22 years old. We try to break the gap through free courses of computational thinking and algorithm design concepts hidden in courses called "Robotics for everyone" or "game development for everyone" using different methodologies describe in the present work, preparing people of all ages to a better understand of computer concepts and improve computer skills.

For this situation with help of our universities we organize a course that involves learning computer science concepts to develop simple programs using C and python language and build robots with Arduino board.

For that, we measure how fast is the adaptive behavior of our students and how fast increase the understanding of these concepts. We hide the concept of Computational Thinking inside of "how to teach computer science using block programming" because computational thinking is the way that any person can interpret the world in a computer and how can extrapolate these concepts to real world. So while we teach about how interpret computer science concepts, we teach how is computational thinking works in our lives [4].

Definitions and techniques

We need to understand important concepts those we use in our research, those concepts are very useful when we try to measure and expose about the progress or difficulties of the students while they are learning new things with enhanced methods through technology. We use Learning Curve to define how they understand new concepts and Adaptive Behavior to define performance and attitude against new concepts.

Learning Curve

We introduce the concept of Learning Curve as the representation how to increases the learning based on experience. also, measuring if the student solve problems better than before times.

Adaptive Behavior

We introduce the concept of Adaptive Behavior as how students accept or reject new concepts. Also, we know that adults, kids and teens have different ways to learn and understand things. The adaptive behavior is notorious when in the learning process they can use examples based on computer science concepts or make jokes with computers or with some new concepts than they never used before.

Script Language

Script Language is a technique used to explain concepts in an very short time with simple examples, script languages are defined as a normal conversation with specific questions, the answers could be any but in all of these answers, there is a concept hidden inside.

Metaphors

Metaphors is used to complete the understanding of the concepts givins another examples based on the previous one, metaphors is very commonly used in the daily activity, in any situation because is the most easy way to explain new things. we use metaphors to exaplin computer science concepts with common examples.

Mind maps

Mind maps is a technique based in drawing a map based on a brain storming with connected ideas like a graph \cite{mind_map_ref}. We use Mind maps to teach about mathematics concepts, programming concepts and algorithm design concepts \cite{teach_mindmaps_programming}.

Block Programming

Block Programming is a technique to encapsule code programming in a simple sentence and the way to programming is just joining the blocks following a flow diagram or main idea. In this research we try to test how efficient is teach computer science concepts and examples using block programming in a first steps and how fast our students can improve skills programming to jump from blocks to code in simple programs like programming sensors or programming games, in this work we encourage to our students and our readers to practice to

thinking in blocks \cite{thinking_blocks} and think how easy could be the learning for future generations topics like IoT or Data Science or Machine Learning with blocks, that research field or develop is part of challenges to learn and understand block programming and how you can add new libraries into a sequence of blocks \cite{challenge_blocks}.

Methodology

We started the classes with the empirical methodology based on Metaphors [5] because or students are novices in the five stage of learning programming skills: novice, advanced beginner, competence, proficiency, and expert [6]. That means, we teach using example of different simple situations in the real life to explain what can we do to interact and which solutions we provide to solve daily problems.

For difference age we use different methodology, as we mentioned above, there is a different way of understanding and retention of information in all of these ages.

Learning Computer Science Concepts

To introduce computer science concepts, we starts with the main question "what is an algorithm ?". To answer that question, we use first technique called script language used in kids [2] in all ages.

At first time adults and Teens understood very fast, kids and tweens take a time to understand the aim of that examples. We use different asks in different ages, simple questions to explain that an algorithm is inside in any situation and in any action.

- ***Script language for kids***

  (Q): “If you want to go to the bathroom, what should you need to say ? How do clean your desk ?"

- ***Script language for tweens***

  (Q): “Have you ever play a game in your computer ?, do you hear about PSP or XBOX ?, Do you know what Mario Kart is it?".

- ***Script language for teens***

  (Q): “Do you know how to solve a linear equation ?, Which rules do you need to follow when you play foot ball ?, Have your ever play guitar ?".

- ***Script language for adults***

  (T): “Have you ever developed something ?, do you hear about Programming languages ?, Do you know what Software means?".

These questions look different but in the background they are very similar. In all categories, they answered "Yes!" and then explained how it works with a sequence of steps. You can see how many time it takes for them.

Implementing a mini Robot

In this section we introduce as the goal of the course the implementation of two mini robots called Otto and Kyo (See Fig. 1). In this part, we use all methodologies to explain about electronic components, the mathematical fundamentals and how they should to program the sensors.

To implement and programming these robots, their movements and their sensors we use Block programming and C language (with the concepts explained above) for this part we use the online platform to simulate the component programming. To programming Otto we use the platform BitBloq and to programming Kyo we use the platform Tinkercad.

At this time, we measure how fast they understand basic electronic concepts and components description and how fast they can imagine a solution to programming using blocks.

We starts with script language then with metaphors and finally with mind maps to get a solution for the problem of how should we programming these mini robots. But in thi part we focus on programming skills and how they can translate their mind maps into a program (blocks or programming language).

We note that there is a improving in programming speed after they understand the logic following in the mind map previously drawn.

Results

As definition of learning curve, we need to verify if the average time of learning in previous works in kids [2] and teens [3] is correct in all new concepts and for computer science we need to add mathematic, physics and electronic components concepts. So we teach and refor this topics with different methodologies, using the formula below we approximate the average time to learning and we compare with the real average time get from the students.

We note that our average time in some cases is a bit less than the approximate time and also is less than the average time mentioned in previous works.

Measuring the learning curve

As definition of learning curve, we need to verify if the average time of learning in previous works in kids \cite{learn_kids_ref} and teens \cite{learn_teens_ref} is correct in all new concepts and for computer science we need to add mathematic, physics and electronic components concepts. So we teach and refor this topics with different methodologies, using the formula below we approximate the average time to learning and we compare with the real average time get from the students.

We note that our average time in some cases is a bit less than the approximate time and also is less than the average time mentioned in previous works.

Where:

K: Number of hours used to understand the first unit (or task).

: Number of hours to understand the unit.
x: Number of the unit.
b: Percent of learning.

So, we have this tables with respective approximations. We take the K value from the first part in script languages in Table I we take b from [7] and x is equal to 4, because is the 4th unit.

TABLE I. TABLE OF AVERAGE TIME TO DESCRIBE AND SOLVE.

How many time takes to learn a new concept with examples ?			
Category	Time to describe the problem	Average Time to solve	Previous work time
Kids	55-65 min	66.486 min	68.64 min
Tweens	52-56 min	57.365 min	56.49 min
Teens	42-52 min	50.934 min	51.32 min
Adults	25-41 min	42.163 min	40.89 min

Conclusion

This work introduces the different adaptive behavior with different methodologies in the learning speed of kids, tweens, teens and adults. We note that tweens and teens have more ability to understand new concepts using games as metaphors. Adults have a strong learning speed to understand new concepts based on past experience. From kids to juniors, they present a fast learning speed, but they forget concepts in a little period of time.

Table I shows the results of manage the learning curve after all the tasks and show us that learning time don't depend of an specific topic, depends of the methodology applied to teach that.

Acknowledgments

This work was supported by grant 234-2015-FONDECYT (Master Program) from CienciActiva of the National Council for Science,Technology and Technological Innovation (CONCYTEC-PERU).

References

INEI, Population index, Lima, Perú, https://www.inei.gob.pe/estadisticas/indice-tematico/population/, Last accessed 2 Feb 2018.

Shahla Gul, Muhammad Asif, Waqar Ahmad and Uzair Ahmad: Teaching Programming: A Mind Map based Methodology to Improve Learning Outcomes. In: International Conference on Information and Communication Technologies (2017).

D. Pérez-Marín, R. Hijón-Neira, M. Martín-Lope : A Methodology Proposal based on Metaphors to teach Programming to children, In: IEEE Revista Iberoamericana de Tecnologias del Aprendizaje (2018).

D. Ginat: On novice loop boundaries and range conceptions, In: Computer Science Education (2004).

A. Robins ,J. Rountree, and N.Rountre: Learning and teaching programming: A review and discussion, In: Computer Science Education (2003).

Sajana Sigde, Technology and Learning Capacity of Children: A Positive Impact of Technology in Early Childhood, Johnson & Wales University - Providence.

Amanda Lenhart, Paul Hitlin and Mary Madden, Teens and Technology, PEW INTERNET & AMERICAN LIFE PROJECT.