

**LEARNING COMPUTATIONAL THINKING  
THROUGH EMBODIED SPATIAL PROGRAMMING  
IN AUGMENTED REALITY**

**Anna Fusté Lleixà**

BEng., La Salle, Universitat Ramón Llull, 2013.  
BComms., Universitat Pompeu Fabra, 2010.

Submitted to the Program in Media Arts and Sciences,  
School of Architecture and Planning,  
in partial fulfillment of the requirements for the degree of  
Master of Science in Media Arts and Sciences  
at the Massachusetts Institute of Technology

*September 2018*

*© 2018 Massachusetts Institute of Technology. All rights reserved.*

Signature redacted

Signature of Author: \_\_\_\_\_

**Anna Fusté Lleixà**

Program in Media Arts and Sciences  
September 2018

Signature redacted

Certified by: \_\_\_\_\_

**Chris Schmandt**

Principal Research Scientist  
Thesis Supervisor

Signature redacted

Accepted by: \_\_\_\_\_

**Todd Machover**

Academic Head, Program in Media Arts and Sciences



# **HYPERCUBES**

Learning Computational Thinking through Embodied  
Spatial Programming in Augmented Reality

**Anna Fusté Lleixà**

## **ABSTRACT**

Computational thinking has been described as a basic skill that should be included in the educational curriculum. Several online screen-based platforms for learning computational thinking have been developed during the past decades. In this thesis we propose the concept of Embodied Spatial Programming as a new and potentially improved programming paradigm for learning computational thinking in space. We have developed HyperCubes, an example Augmented Reality authoring platform that makes use of this paradigm. With a set of qualitative user studies we have assessed the engagement levels and the potential learning outcomes of the application. Through space, the physical environment, creativity and play the user is able to tinker with basic programming concepts that can lead to a better adoption of computational thinking skills.

# **HYPERCUBES**

Learning Computational Thinking through Embodied  
Spatial Programming in Augmented Reality

**Anna Fusté Lleixà**

This thesis has been examined by the following reader:

Dr. Pattie Maes  
Professor of Media Technology, MIT Media Lab.

**Signature redacted**

MIT Program in Media Arts and Sciences.

# HYPERCUBES

Learning Computational Thinking through Embodied  
Spatial Programming in Augmented Reality

**Anna Fusté Lleixà**

This thesis has been examined by the following reader:

Amit Pitaru  
Creative Director, Google Creative Lab.  
CoFounder, School For Poetic Computation.

**Signature redacted** .

Google, SFPC.



## **ACKNOWLEDGEMENTS**

These have been two incredible years that have deeply inspired me; not because of the place or the projects around me but because of the amazingly talented people that have influenced and helped me evolve in such thriving environment as the MIT Media Lab.

First and foremost, I would like to express a special thanks of gratitude to my mentor, Chris Schmandt. Thank you for accepting me into the group. Your help and feedback have allowed me to make the right decisions to reach a higher quality of research. Thank you for always being straight forward and for giving me so much freedom to explore. Thank you as well for showing me the desert and the mountains and for believing in me and my ideas every step of the way.

I would also like to thank my two readers Pattie Maes and Amit Pitaru. Pattie, your inspiring mentorship and your guidance have given me the perfect criticism to perfect my research and ask the right questions. Amit, thank you for giving me the opportunity to spend some time at the Creative Lab where all this thesis was originated. Your feedback has inspired me to look further and beyond.

This thesis project also had the opportunity to be evaluated through a set of qualitative user studies as a deployment in a non-lab setting, thanks to a special group of people. I would like to thank first Luba Elliot, Tracy Harwood and Tina Barton for inviting me to the Leicester Art-AI Festival and for allowing a set of workshops with the most amazing kids. It was a pleasure to attend and witness their reactions to my work. I would also like to thank Parts & Crafts for having me during their Girls Invention Week and allowing the children to play with the project. Finally, I would like to thank the MIT Museum for being interested in my work and letting me perform a workshop at their IdeaHub program where I had the chance to test the application and record the families playing with it.

A massive thank you to all my lab friends that have been there during fun times and hard times these past two years. To Javi and Oggi, for your advice in my research but more importantly for making me laugh, for your constant visits to my office, for teaching me about affective computing and personalized machine learning and for supporting me during my stay at the lab.

To Pedro, Nikhita and Chrisoula. 'Papa', thanks for always being there, for giving me plenty of good moments and helping me through the struggle. Nikhita thanks for all the support, the 'calm down' talks and for inspiring me with your amazing design and art skills. Chrisoula you have been an inspiration and I have admired you since I met you. Your talent and your resilience will take you very far. Thank you for being my rock.

To my best friend Judith. You not only give me motivation for my research work but you are also an inspiration in my life. We both started this research together and it is thanks to you that this project has been possible. Thank you for believing in me and for inspiring me in so many different domains in my life.

To my grandparents, my uncles, aunts and cousins. My work reflects the values and principles that I have gained during my entire life and most of them are influenced by all of you. A big thank you for shaping all these values and for supporting me in every decision even if it means being far away.

To my sister Laura and my parents Teresa and Jordi. Laura, you are and always will be my soul mate and my other raindrop. No matter how far you are you will always inspire me to thrive. Mama, thank you for your encouragement, for guiding me in my research and for all the love you give to me. Papa, thank you for believing in me in every step that I have taken and for all your excitement and admiration towards me. I look up to the three of you. You have made of me the person that I am today and I deeply thank you for that.

And finally, a tremendous thank you to my husband, Jordi. Thank you for everything you do for me every day. Your constant support and the love you give to me keep me pushing forward and being creative in all the work that I embark on. You inspire every single one of my movements and decisions and give me all the happiness in this life. You give me all I need and I thank you for that.

# TABLE OF CONTENTS

<b>1</b>	<b>INTRODUCTION</b>	<b>9</b>
1.1	Motivation	9
1.2	Research goals	11
1.3	Thesis structure	12
<b>2</b>	<b>BACKGROUND</b>	<b>14</b>
2.1	<b>About experiential learning and computational thinking</b>	<b>14</b>
2.1.1	Towards a constructionist theory of learning	14
2.1.2	Learning through play	18
2.1.3	Computational thinking	21
2.1.4	Learning about programming	26
2.2	<b>About space, tangibility and Augmented Reality</b>	<b>29</b>
2.2.1	Space and embodiment for STEM	29
2.2.2	Tangibility and materiality	33
2.2.3	Augmented Reality	37
<b>3</b>	<b>EMBODIED SPATIAL PROGRAMMING</b>	<b>42</b>
3.1	<b>Humans and our spatial mental models</b>	<b>42</b>
3.2	<b>Augmented reality, the body and space</b>	<b>48</b>
3.2.1	Do not gesture nonsense in space	48
3.2.2	Towards a graspable future	52
3.3	<b>Authoring to own the medium</b>	<b>55</b>
3.4	<b>Embodied spatial programming for computational thinking</b>	<b>57</b>
<b>4</b>	<b>HYPERCUBES</b>	<b>61</b>
4.1	<b>Overview</b>	<b>61</b>
4.1.1	HyperCubes outline	61
4.1.2	DIY culture and maker culture	61
4.1.3	Tangibility and raw materials	63
4.1.4	Mobile device as an accessible medium	63
4.1.5	Embodied Spatial Programming in AR	64
4.1.6	Paper Cubes and HyperCubes	64
4.2	<b>Application design</b>	<b>66</b>
4.2.1	Learning computational thinking with HyperCubes	66
4.2.2	The cycle as a programming loop or main function.	80
4.2.3	Cubes and Parameters as Functions and Variables	81
4.2.4	Cubes as Programming Functions and Events	86

<b>4.3</b>	<b>HyperCubes as an authoring tool for creativity</b>	<b>88</b>
4.3.1	Playfulness and creativity for learning	88
4.3.2	Low floors and high ceilings	88
4.3.3	Collaboration in space	89
<b>4.4</b>	<b>System overview and implementation</b>	<b>90</b>
4.4.1	Design and UX	90
4.4.2	Main Software Architecture	94
4.4.3	Augmented Reality technologies	96
<b>4.5</b>	<b>Evaluation</b>	<b>98</b>
4.5.1	Pilot study	98
4.5.2	Qualitative user study	104
<b>5</b>	<b>CONCLUSIONS</b>	<b>111</b>
<b>6</b>	<b>FUTURE WORK</b>	<b>114</b>
<b>7</b>	<b>REFERENCES</b>	<b>116</b>
<b>8</b>	<b>TABLE OF FIGURES</b>	<b>121</b>
<b>9</b>	<b>APPENDIX</b>	<b>124</b>



# **HYPERCUBES**

Learning Computational Thinking through Embodied Spatial Programming in Augmented Reality

## **1 INTRODUCTION**

### **1.1 Motivation**

Since Logo (Papert, 1980), the first programming language to teach programming skills to children, a great number of tools have been developed to improve computational literacy. Computational thinking is now considered a basic skill that should be included in the educational curriculum in schools (Lockwood & Mooney, 2017). It has also been regarded as a learning framework that can incorporate other basic subjects such as mathematics, engineering or science (Wing, 2006).

The majority of the tools that are being developed for computational thinking are screen-based online tools such as Scratch (Resnick, et al., 2009) and Alice (Cooper, Dann, & Pausch, 2000). In making use of these tools, the child engages in an activity in front of a screen that isolates her from her physical surroundings.

These experiences make no use of space and embodiment, which are regarded as basic and crucial to the learning process by well-known cognitive and psychological theories (Antle, 2009; Newcombe, 2017; Wai, Lubinski, & Benbow, 2009). As Antle (2009, p. 27) explains, “An embodied approach suggests that humans should be considered first and foremost as active agents rather than as disembodied symbol processors.” She also highlights “The importance of understanding the role of action and the environment in the development of children’s thinking skills.”

Parallel to this, Augmented Reality (AR) has gained a lot of attention in recent years with improved hardware and software and new devices for the masses. New gestural interfaces have been designed that do not take advantage of the physicality of our environment and our human spatial mental models. AR is a very powerful technology that lets us make use of

the flexibility of the computational medium without losing the physical texture of our environment. We can still interact with real objects instead of just staring at a flat screen. We can grasp and hold things and we can have a digital layer that provides an information bridge to our mental constructs.

To this end, we propose the concept of Embodied Spatial Programming as a framework to learn computational thinking. The principles of Embodied Spatial Programming are based on the idea of using physical tokens and objects to program digital interactions in space. AR's capability to bridge the physical and virtual worlds makes it a suitable technology to apply Embodied Spatial Programming principles to the interaction design on new educational interfaces and applications.

We also present HyperCubes as an example application of Embodied Spatial Programming. HyperCubes is an AR platform for children to create and learn by spatially orienting handmade paper cubes. HyperCubes sets a successful precedent for the advantages of applying Embodied Spatial Programming to an AR creation and educational tool.

With new technologies that make use of space and the physical environment of the user, such as AR, we can create improved interactions to teach computational thinking to new generations, using the full agency of their bodies, without losing the benefits of the computer as a medium.

## 1.2 Research goals

The main research goals of this thesis are the following:

- 1. Propose a new interaction framework for learning computational thinking that makes use of space and the physical environment of the user.**
- 2. Develop and validate this new interaction model with an Augmented Reality authoring platform using physical paper cubes in space with the aim of teaching computational thinking concepts in a creative and playful environment.**

The first research goal focuses on the idea of designing a new interaction framework based on a body of work related to spatial thinking and user interaction with the environment. Considering the background work and previous interaction theories, we want to propose an improved interaction model that makes use of space and the physical environment in order to design programming interfaces.

The second research goal is focused on trying to consolidate this model by designing and validating an AR authoring platform that makes use of the principles of this interaction model. The main intention is to have a solid project based on our theory to be able to test it on a non-laboratory setting where we can evaluate and get some feedback on the design advantages or disadvantages of the proposed model.

The evaluation performed for the project is a qualitative evaluation adjusted to the thesis scope. This qualitative evaluation has given us a sense of the positive and negative impact that this interaction model can have on the user. In order to demonstrate learning outcomes, we will need to perform a quantitative evaluation and more qualitative assessments as discussed in the Future Work section.

### **1.3 Thesis structure**

The thesis starts off with an overview of the related work and theories that have been explored in the past. This background chapter is split in two sections.

The first section is focused on two main points: giving a background about psychology theories related to learning and defining computational thinking as a key subject that should be included in the educational curriculum.

The first subsection goes through well-known cognitive development psychology theories such as the Constructivist theories of Jean Piaget or the Constructionist theories of Seymour Papert. The second subsection talks about the creative learning approach of Mitchel Resnick and gives some examples of this approach such as the Explorable Explorations platform. In the third subsection, computational thinking is explained and examples of its application are mentioned. The last subsection dives into the concept of learning about programming, what it means and why is it important.

The second section of the background is focused on space, tangibility and Augmented Reality. It starts with a subsection on Visual Programming as the seed to using space for our everyday interfaces. It later goes on to explain cognitive theories that put spatial thinking as a key framework for learning in STEM fields. It then explains tangibility and materiality as two key aspects when designing interfaces. Finally, Augmented Reality is presented as a recently boosted technology that can help leverage all the cognitive spatial and tangible models that humans already have.

The second chapter of the thesis proposes Embodied Spatial Programming as a framework to design new interfaces for learning about programming. The first section talks about the spatial mental models that humans develop and why and how we can take advantage of them. The second section talks about Augmented Reality as a powerful technology for leveraging these spatial mental models and it presents good and bad practices that have already been seen using this technology. The third section talks about the need for developing tools native to this new spatial medium and the importance of these creation tools. The final section dives deeper into the concept of Embodied Spatial Programming, its definition and uses.

The third chapter is focused on HyperCubes, an example application that makes use of Embodied Spatial Programming. This chapter contains a first overview section with an introduction to the application, why it was developed and what principles were used in its design. The second section explains the application design in more depth and explains how the application works. The third section talks about the system implementation and the technologies used for the development of the application. The last section explains the evaluation performed of the application and gives some initial qualitative results extracted from the interactions in a pilot study and in a user study.

Finally, the thesis presents a Conclusions section with final remarks, lessons learned and the possible improvements for the project. After the Conclusions, a Future Work section talks about the potential for the application and what the next steps are in order to improve it and take it further.

## **2 BACKGROUND**

### **2.1 About experiential learning and computational thinking**

The educational curriculum has evolved through the last decades towards learning-by-doing methodologies and experiential learning approaches. Different psychology theories have tried to find a proper means of improving the cognitive development of a child considering different maturity stages and levels. Technology has played a key role in this evolution, giving place to a set of good practices that take advantage of the computational medium but also a vast number of methodologies that lack some of the essential ingredients for the proper cognitive development of a child. It is worth reviewing all these theories and approaches in order to design improved interfaces that can help advance the educational curriculum.

#### **2.1.1 Towards a constructionist theory of learning**

##### ***Piaget's theory of cognitive development***

“Each time one prematurely teaches a child something he could have discovered himself, that child is kept from inventing it and consequently from understanding it completely.”  
- Jean Piaget

In order to build a framework for a learning environment, one needs to fully understand all the factors, context and processes that will affect a child during the course of her cognitive development cycles. There has been a vast body of work exploring the mental processes of the development of a child. Based on the work of Jean Piaget, the father of constructivism, others focused on a more practical approach such as the constructionist theories of Seymour Papert or the creative learning approaches of Mitchel Resnick. It is worth reviewing all these theories in order to be able to design proper systems founded on the psychology of the human and from the ground up.

Jean Piaget (Piaget, 1971) is considered one of the main pioneers of constructivism in the psychology field. He introduced his theory of cognitive development based on the idea that

the development process is conformed by a cycle of steps. These steps are based on experience and experimentation with actions and they build up after each cycle. He claimed that children had an active role in the learning process and that through experimenting and play, they would build upon the knowledge they had already acquired.

Piaget presented four different stages in the development of a child:

- “Sensorimotor” stage: birth to 2 years
- “Preoperational” stage: ages 2 to 7
- “Concrete operational” stage: ages 7 to 11
- “Formal operational” stage: ages 12 and up

In the first “sensorimotor” stage the child needs to be in contact with her physical surroundings. Piaget assumed that the mind in a child developed endogenously until the infant had the cognitive capacity to assimilate language. In the “preoperational” stage, the child begins to learn to speak, however, she is not able to reason or apply logic. It is a stage that he considered related to play and pretending. In the “concrete operational” stage, children can think logically, but their reasoning is limited to the physical possibilities. Finally, in a more advanced stage, the “formal operational” stage, the child can deal with symbols and abstract thinking.

Years after, researchers would start realizing how these stages should not be treated as sequential but rather parallel ongoing events. The acquisition of sensorimotor skills and the principles of cognitive development were presented as events that should be intertwined in order to boost the learning processes.

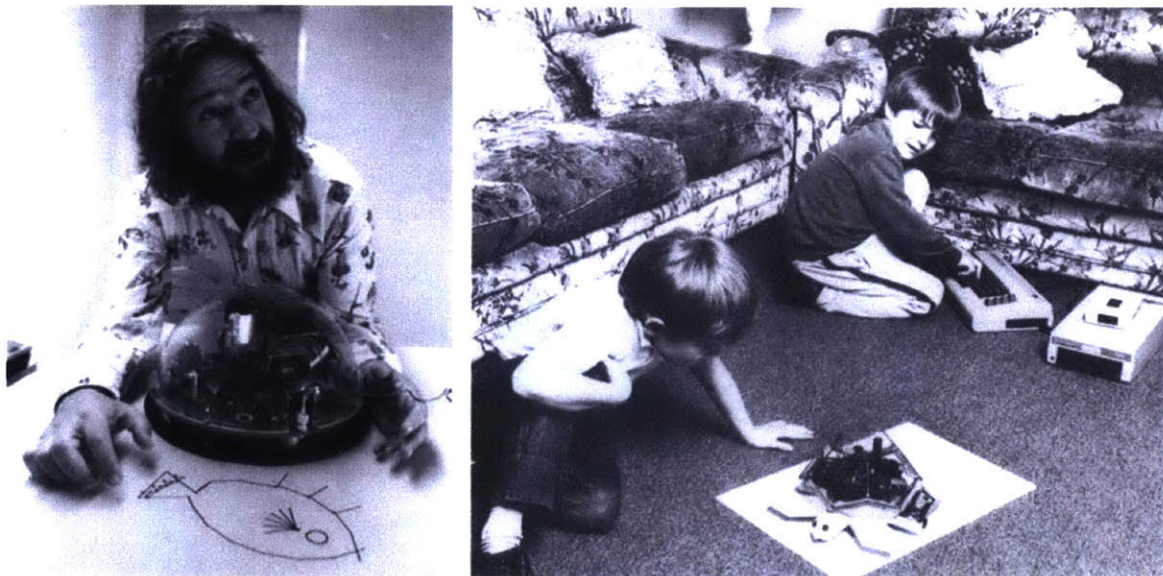
### ***The constructionism of Seymour Papert***

Following Piaget’s theories of constructivism, Seymour Papert developed what he called constructionism. Constructionism is a ramification of constructivism and is based on the idea that children learn by creating mental models in their brains drawing from experience and action. Discovery learning is key and it builds upon knowledge previously acquired. Papert was a big advocate for project-based learning. He was a mathematician, computer scientist and educator and one of the pioneers of artificial intelligence. He spent most of his career

working on learning theories and used new technologies as a playground for the Constructionists learning methods.

The principles of constructionism say that the learning process is more effective when students are engaged in making and interacting with tangible objects in the physical environment. As Papert (Papert, p. 1, 1987) defined the concept himself:

“The word constructionism is a mnemonic for two aspects of the theory of science education underlying this project. From constructivist theories of psychology, we take a view of learning as a reconstruction rather than as a transmission of knowledge. Then we extend the idea of manipulative materials to the idea that learning is most effective when part of an activity the learner experiences as constructing a meaningful product.”



*Figure 1. From left to right, Seymour Papert with the Logo Turtle<sup>1</sup> and two children playing with it<sup>2</sup>*

Papert strongly advocated the use of the computer in the classroom. Constructionism was initially used for teaching mathematics, however Seymour Papert started putting emphasis

---

<sup>1</sup> From [Matematicamente.it](http://Matematicamente.it) (2014)

<sup>2</sup> From Catlin, Dave (2016) “My personal tribute to Seymour Papert”. Go Magazine



on what he called computational thinking as a new way of engaging with the new era of technology. Several programming languages and environments have been developed as platforms for Constructionist learning and computational thinking. Papert himself developed the language Logo which introduced the turtle concept [Figure 1] to elementary schoolchildren. Other languages and platforms following the constructionist principles are SmallTalk (Kay, 1996), Etoys (Kay, 2005) and Scratch (Resnick, et al., 2009).

## 2.1.2 Learning through play

### *Lifelong Kindergarten*

Mitchel Resnick, professor at the MIT Media Lab, starts his book *Lifelong Kindergarten* (Resnick, 2017) by giving the example of Chen Jining, the president of Tsinghua University, the leading engineering university in China. He explains how Chen Jining was worried that the Chinese students, although having very high standards and good grades, were not really prepared for the needs of our constant changing and evolving society. The students knew the theory, they knew how to solve math problems, but they didn't have the ability to think creatively, to ask their own questions, to pose their own queries and solve them. Resnick, influenced by the constructionist theories of Papert, talks about the need to forge a new generation of students that are curious, that think further and can formulate new challenges built upon the knowledge that they acquire. In order to prepare students for this ever-changing future, we need to carve creative thinkers.

Resnick runs his group Lifelong Kindergarten at the MIT Media Lab, where he works towards a vision of the educational curriculum much more open to creativity and free play. He talks about the creative process as a "Creative Learning Spiral" [Figure 2] significantly related to the cognitive development theories of Piaget and the constructionist theories of Papert. In his "Creative Learning Spiral" there are five different stages: "Imagine", "Create", "Play", "Share" and "Reflect". It is defined as an infinite cycle that builds up at each iteration:

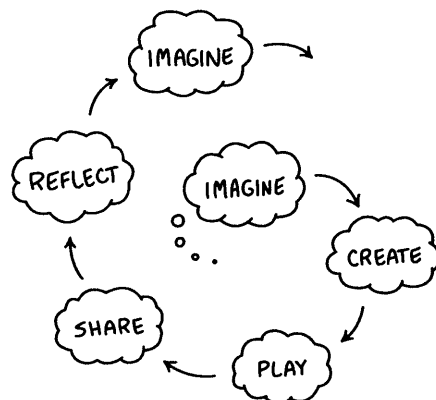


Figure 2. Creative Learning Spiral by Mitchel Resnick<sup>3</sup>

---

<sup>3</sup> From Resnick, Mitchel (2017). Courtesy of the author

At the Lifelong Kindergarten group, Resnick has developed a framework for creative thinking based on four basic principles [Figure 3]: “projects”, “passion”, “peers” and “play”. Resnick explains how creating “projects” is essential for the Creative Learning Spiral and highlights some of the examples in the Scratch community where users create projects constantly that help them develop their skills and creativity. As Piaget said: “Children learn best when they are actively engaged in constructing something that has personal meaning to them”, that is why “passion” is fundamental for the creative process. “Peers” refers to the collaboration and sharing of projects with others. As Resnick states: “Creativity is a social process”. Finally, the last core is “Play”. Playful experimentation is key in order to successfully thrive in the creative process.

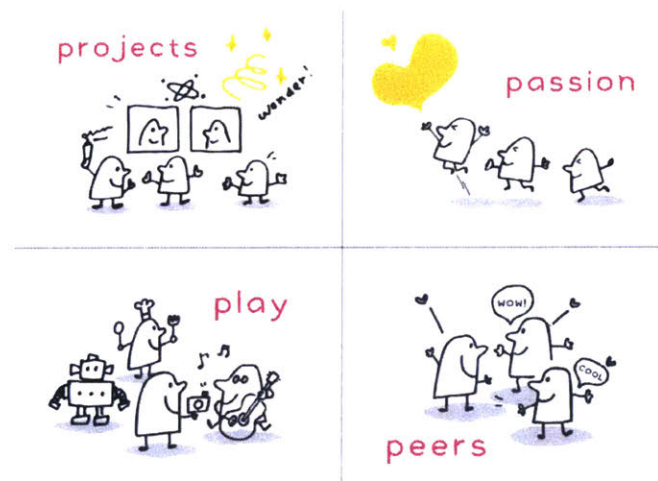


Figure 3. *Projects, Passion, Peers and Play* by Mitchel Resnick<sup>4</sup>

### **Explorable Explanations**

This playful experimentation is key for the learning experience and has been included as a guiding principle in many learning environments. An interesting example of using play for the learning process is found in Explorable Explanations. Explorable Explanations is defined as a community of artists, coders and educators that develop interactive projects to learn through play. As Bret Victor explains (Victor, 2011), Explorable Explanations is an “umbrella project

<sup>4</sup> From Resnick, Mitchel (2017). Courtesy of the author

for ideas that enable and encourage truly active reading”. Bret Victor talks about active readers as the ones that “ask questions, consider alternatives, question assumptions, and even question the trustworthiness of the author”. This definition aligns considerably with the new generation of creative thinkers that Resnick talks about. In Explorable Explanations, the user can learn about a topic by engaging with an interactive platform or project that allows for play and exploration. After developing some of the Explorables in the platform, Nicky Case (2018) talks about four design patterns for teaching through play that he has been able to extract from the Explorable Explanations experiments. These design patterns are the following ones:

- “Puzzle it out”: Case defends the idea that puzzles make the user think. And by thinking the user can learn and assimilate new concepts.
- “Place your bets”: let the user future-cast what the solution will be and compare her solution to the correct answer so she can see how her expectations match or don’t match reality.
- “Role play”: this design patten helps not only using empathy but making the user think about ideas, reasoning and pondering on issues and situations.
- “Sandbox mode”: this design pattern is based on giving the user a simulation to play with. Case highlights the fact that this pattern needs a ground-up approach where the user starts off with a very basic simulation and everything leads to the whole sandbox.

These patterns define a practical framework guided by the principles of learning through play that Resnick defends. These practical guidelines are some of the basic foundations that should be considered when designing new learning environments for playful experimentation.

## 2.1.3 Computational thinking

### *Origins and definition*

“If children became fluent in thinking in the medium of the computer; If programming was a form of basic literacy like reading and writing, they’d become adults with new forms of critical thought.”

- Alan Kay

The term computational thinking was first introduced by Seymour Papert in his book “Mindstorms: Children, computers and powerful ideas” in 1980. Since then, there has been a significant debate on what exactly computational thinking is, how we can benefit from it and why we should learn about it.

“Mindstorms” marked a turning point in the educational landscape. Teachers and educators started introducing computers in the classroom as a new creative pedagogical tool. The resilience of the computational medium offered limitless possibilities for children to explore new ideas while boosting their cognitive development. Papert invested much of his time working on his platform Logo, that he designed by applying computational thinking principles for children. Logo allowed children to program using a high level language to draw shapes on the screen [Figure 4] or drive a robotic turtle that drew them on the floor.

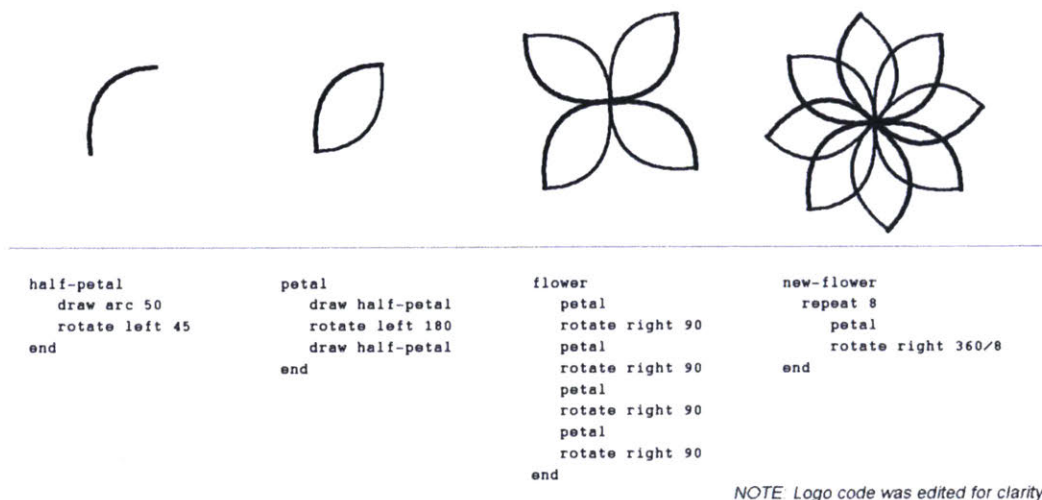


Figure 4. An example of a Logo problem to draw a flower<sup>5</sup>

<sup>5</sup> From Ferster, Bill (2013). Adapted from Papert (1993). Retrieved from flickr.com

As Papert introduced it, the term computational thinking not only is a way of conceiving and unraveling the computational medium. It is also a way to enhance the understanding of our reality and the way we interact with it. It is a framework for our mind to help us in being better problem solvers and creators.

### ***Skills and approaches***

Computational thinking defines a set of skills and approaches that go beyond basic coding skills. Most of these skills and approaches can be found in a regular computer science pedagogical curriculum but can be applied in any situation where a person may need to learn or carry out a task. These are the skills and approaches [Figure 5] that I consider to be more significant and that I put together based on the “Mindstorms” book and other sources (Resnick, 2017; Bers, 2017). The skills that constitute the basics of computational thinking are the following:

- Logic: we are able to reason to solve a problem, situation or procedure.
- Decomposition: we have the ability to break problems down into parts.
- Pattern recognition: we can find similarities around us and predict events.
- Abstraction: we are able to work with broader ideas and symbols.
- Algorithms: we are capable of developing abstract computation sequences.
- Simulations: we can create rules and put them to practice in parallel environments.

All these abilities can be mastered by taking some basic approaches that boost their strength such as tinkering, creating, persevering, debugging or collaborating. Modern educational platforms have taken into account all these approaches to build systems that enhance the child’s cognitive development (Cooper, Dann, & Pausch, 2000; Lockwood & Mooney, 2017).

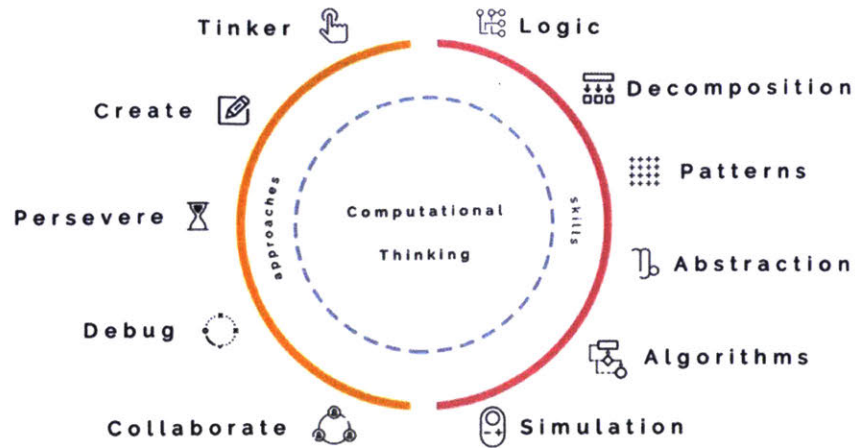


Figure 5. Computational thinking skills and approaches

## ***Beyond Papert***

“Children learn best when they are actively engaged in constructing something that has personal meaning to them – be it a poem, a robot, a sandcastle or a computer program.”  
 - Seymour Papert

Logo (Papert, 1980) was the first programming language created for teaching Computational Thinking concepts. Since its creation, numerous other platforms and programming languages have been developed with the same objective (Cooper, Dann, & Pausch, 2000; Kay, 2005). Some of these platforms draw from Papert's constructionist learning approach, such as Scratch (Resnick, et al., 2009), a free programming language and online community where users can create their own interactive stories, games, and animations [Figure 6].

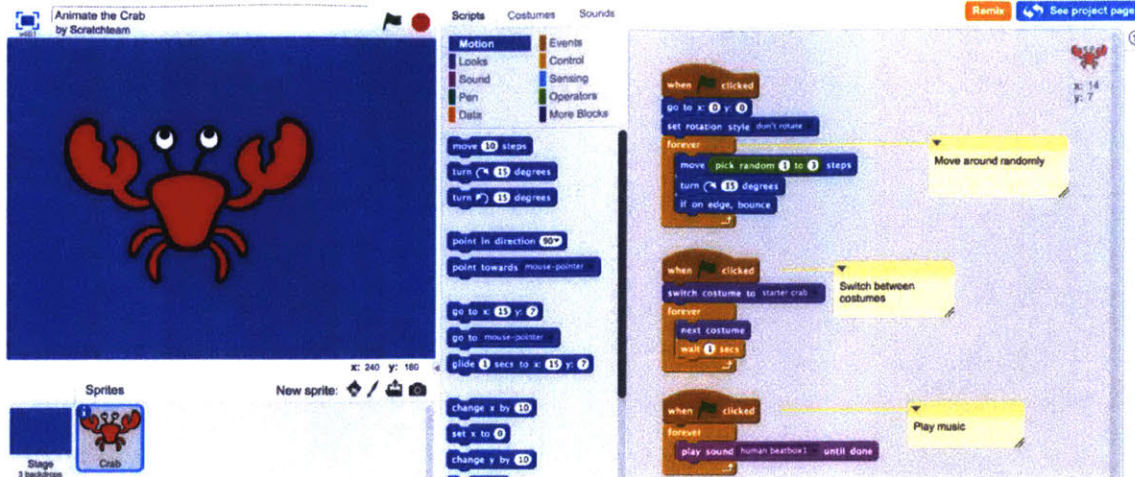


Figure 6. Scratch Interface<sup>6</sup>

Scratch and many others make use of visual programming. Alice (Cooper, Dann, & Pausch, 2000), AgentSheets (Repenning, 1993), MIT App Inventor (Pokress & Dominguez Veiga, 2013), CS Unplugged (Bell, Alexander, Freeman, & Grimley, 2009) or the LEGO Mindstorms (Klassner & Anderson, 2003) are just some of the platforms focused on teaching coding to children using computational thinking concepts [Figure 7].



Figure 7. Computational thinking pedagogical platforms<sup>7</sup>

Likewise, during these past decades several research studies and experts have claimed how computational thinking is essential for the development of children. Since Papert introduced the term, the research community has been pushing for the inclusion of computational thinking into the educational curriculum. However, although a lot of work has been done

<sup>6</sup> From Scratch.mit.edu (2018)

<sup>7</sup> From Scratch.mit.edu / Appinventor.mit.edu / Csunplugged.org / Lego.com / Agentsheets.com / Alice.org (2018). Logos retrieved and adapted.



around the world in many different educational contexts, the work relating to computational thinking incorporated in the classroom is still in its infancy (Lockwood & Mooney, 2017).

The main reason lies in the fact that we need children to engage and play with their environment. And having them in front of a screen typing lines of code is not ideal for their own development. The vast majority of these educational tools are developed in 2D environments where the child is interacting solo with a screen and not making use of the space around her. Others such as the LEGO Mindstorms, are based on robotics and mechanisms and the programming gets limited by the physical affordances.

There are many open paths and opportunities to explore and there is also plenty of room for improvement in the educational arena. Computational Thinking conforms a framework that not only is good for learning about the computing medium but can also become a solid infrastructure for learning in other areas such as Mathematics, Physics, Literature or Art.

## 2.1.4 Learning about programming

Computational Thinking is much more than knowing how to code. Computational thinking is a way of understanding and interacting with the world. Computational thinking does not mean “thinking like a machine”, the same way as learning about programming does not mean learning how to code. As Bret Victor says (Victor, 2012): “A person is not a machine, and should not be forced to think like one”. Learning about programming means being able to understand certain coding constructs that will boost our way of approaching a problem and finding a solution. As we have already mentioned, Papert put a lot of effort in developing Logo, a programming language intended for educational purposes and focused on learning about programming. Several other pioneers in the field have presented other platforms and languages as well as relevant research on the matter of learning about programming.

One of the most noticeable examples was developed by Alan Kay, a computer scientist who has dedicated great part of his life to the development of educational technology. Kay designed the Dynabook concept, in 1968 (called The KiddiComp back then), as “a personal computer for children of all ages” (Kay, 1972). The Dynabook concept was conceived way before the first tablet was released to the market. He also developed various programming languages and educational platforms such as Etoys. Etoys is an authoring environment and visual programming system for children built in Squeak, an open-source implementation of SmallTalk. Etoys served as an inspiration for the development of Scratch later on. Kay didn't envision Squeak and Etoys as tools to learn how to code. He designed them as platforms for a greater purpose; platforms to learn about programming as an umbrella for the assimilation of many more subject areas such as physics or mathematics.

Kay gives an example of a typical 5th grade math-science project sequence in his research note “Squeak Etoys, Children & Learning” (2005):

- *draw objects (such as cars)*
- *write scripts to move them, have them follow edges, etc. Every fifth frame of video*  
*Dropping objects*
- *think about speed and acceleration by leaving trails*
- *observe moving objects in the real-world: dropping weights, etc.*
- *get detailed records of the drop using video cameras*

- *look at every 5th frame and measure the speeds*
- *compare the difference in speed between each frame*
- *see that the differences (the acceleration) is constant*
- *write a script to move a simulated ball with constant accelerated motion*
- *find the constant that will match up the simulation with the real-world*

It is clear how the student not only learns about scripting but explores and tinkers with a wider range of concepts and transversal competences such as physics, mathematics, writing or documenting. Kay has done substantial work on the area of visual and interactive programming. He created SmallTalk as an educational programming language that was object-oriented, dynamically typed and reflective tailored to the new 'human-machine symbiosis' era [Figure 8].



Figure 8. SmallTalk being used on an Alto computer, by children<sup>8</sup>

All these characteristics are basic to design a technological platform for learning. As Bret Victor points out (Victor, 2012), a platform for learning about programming should aim for these two main goals:

- *Support and encourage powerful ways of thinking*
- *Enable programmers to see and understand the execution of their programs*

---

<sup>8</sup> From Computer History Museum (1970) © PARC (Palo Alto Research Center, Incorporated)

A system to learn about programming should be reflective; affording the ability to examine, introspect and modify its own behavior at runtime. And this, not only applies to a platform to learn about programming but also any other natural interface. A digital drawing program should also be reflective, such as Sketchpad, the first graphical computer-aided design program or ChalkTalk (Perlin & Nunes, 2017), a conversational and performative sketch language [Figure 9]. ChalkTalk was designed and implemented by Ken Perlin. ChalkTalk is an interactive blackboard where the user can draw glyphs and they come to life. It is meant to be used as an educational and conversational platform to teach and discuss topics.

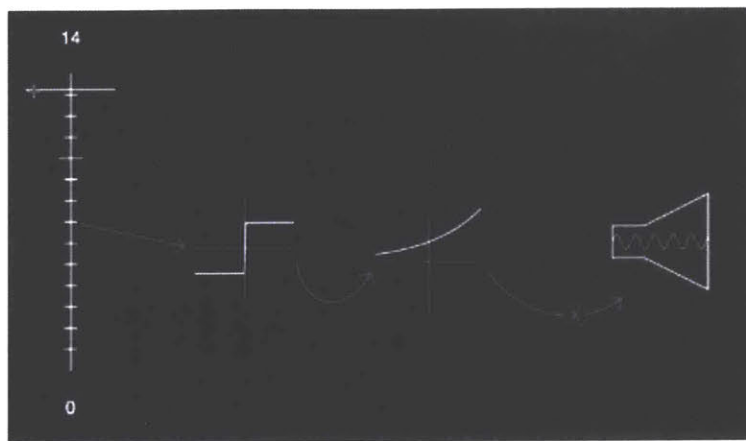


Figure 9. ChalkTalk sketch example<sup>9</sup>

Michael Nielsen put it in these words when examining the ChalkTalk system (Nielsen, 2017): “There is a sweet spot for internalizing operations: they must be performable on a timescale of no longer than a few seconds. That is, there is a natural speed of thought, and what we internalize must match this speed.”

The best way of learning what gravity is, is to release an object in mid-air and observe what happens next. The same applies to programming. If we want someone to learn programming constructs we need a system that can react instantaneously to the user actions the same way as objects in the physical world react instantaneously to gravity.

---

<sup>9</sup> From Sigal, Jason (2018). Retrieved from [jasonsigal.cc](http://jasonsigal.cc)

## **2.2 About space, tangibility and Augmented Reality**

### **2.2.1 Space and embodiment for STEM**

#### ***Visual programming***

The research community has put emphasis on the importance of space and embodiment used in educational methodologies for STEM fields. The body and the relationship with the environment is not only essential for the development of motor skills but it can greatly benefit the understanding of more abstract concepts such as mathematics or physics. We humans exist in a three dimensional world and our brain takes this 3D environment as a reference. This reference has served in the past and has been considered as a powerful conduit to teach science and technology concepts to children. The use of spatial principles to make an interface more usable and instinctive is not something new in technology systems. This idea has been applied to computer interfaces since the start of the computing era. Spatial information and awareness has been used as a tool for designing interfaces better adapted to our human nature.

One of the clear examples of this is visual programming. Visual programming is a powerful and prevalent programming paradigm that has been evolving since the 60s. It makes use of visual references and blocks arranged in 2D space in order to program a regular sequence of commands. Visual programming is the first programming paradigm that leveraged our spatial mental schemes in order to design a better interface for a sequential task. The first visual programming interface was created in 1968. Back then Tom Ellis created GRAIL (Ellis, Heafner, & Sibley, 1969), GRaphical Input Language, where users draw flowcharts in order to write software [Figure 10].

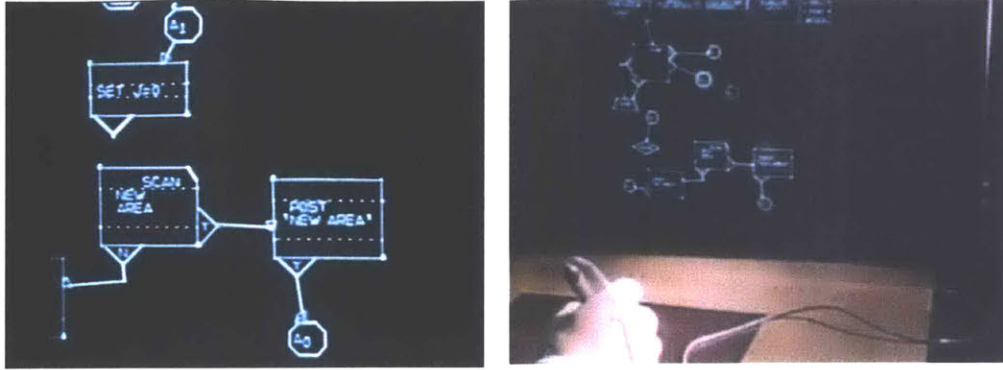


Figure 10. GRAIL Interface (1966)<sup>10</sup>

Since then, plenty of other programming platforms and environments have been developed, specially for learning about programming using spatial principles. An example of this is AgentSheets (Repenning, 1993), a domain-oriented visual programming system; WWW (WWW group, 1998) a node based programming platform to program graphics; or the Reality Editor (Heun, Hobin, & Maes, 2013), a tool to program smart objects in Augmented Reality.

### ***From action to abstraction: ‘Doing with Images makes Symbols’***

Kay worked on trying to apply the principles of visual programming to learning environments for children. Inspired by the theories of Seymour Papert and Jean Piaget, Kay tried to understand how spatial notion was essential when abstracting from reality and trying to think in the computational medium.

If we look back at Piaget’s theories of cognitive development, he explained how there are different stages to the development of a child. A first *sensorimotor stage* where the child needs to be in contact with his/her physical surroundings; a more advanced stage where the child can recognize patterns and the final stages where the child can deal with symbols and abstract thinking. Years after, researchers started realizing how these stages should not be treated as sequential but rather parallel ongoing events.

To make a simile with Piaget’s theories, Alan Kay (Kay, 1987) talked about the idea of ‘Doing with Images makes Symbols’ [Figure 11]. As mentioned before, Kay is a computer scientist and he is well-known for his work on visual programming languages. He tried to explain how

---

<sup>10</sup> From Okamoto, Russell (2016). Retrieved from Medium.com

the learning process would become much richer if we combined Piaget's learning stages altogether. If children could make use of space, the physicality of their environment as they can recognize patterns and understand symbols, the learning process would be much richer and faster.

## DOING with IMAGES makes SYMBOLS

*From Piaget to Alan Kay*

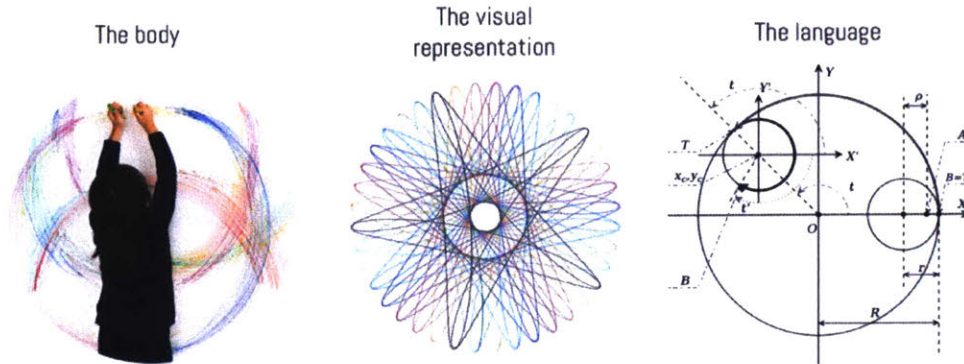


Figure 11. *Doing with Images makes Symbols*<sup>11</sup>

A lot of the inspiration for Kay's work came from Papert's theories. Papert talked about the idea of body-syntonic learning. He realized how children were able to reason using their bodies and create programs for the Logo Turtle by making a simile with their movements in space. Embodiment was key in understanding some of the programming constructs in the Logo environment. He described how concepts such as variables or recursion could be more naturally internalized by using the body and space. For example, a child tried to program the turtle to make a circle. The child would then see how if she moved forward one step and rotated  $x$  degrees and then repeated that sequence over and over she would be able to complete a circle in space.

<sup>11</sup> Adapted from Kay, 1987. "Doing with images makes symbols"

## ***Spatial thinking for STEM fields***

Nora Newcombe is a researcher in cognitive development and cognitive psychology. Her body of work focuses on spatial reasoning and the development of spatial representation. She talks about the importance of spatial thinking for STEM fields, and how it relates to episodic memory (Newcombe, 2017). She conducted several research studies where she found that children that play with blocks and puzzles develop a better spatial reasoning ability. And this spatial reasoning ability is very beneficial for STEM education. Mathematics or physics are great examples where spatial reasoning is often essential to understand abstract concepts. Newcombe talks about more recent studies where there is evidence of the causal links between spatial thinking and STEM fields. She highlights how these studies “show that high spatial ability early on as a child can predict later interest and success in STEM fields”.

Newcombe presents a view of spatial reasoning inspired by Piagetian theories but opposing some of their principles. She talks about the fact that infants are born with spatial coding abilities that evolve during development and life. This view contradicts the Piagetian view that focused more on the idea that children only developed spatial reasoning in the late stages of childhood.

This spatial information encoded in the child’s mind since birth and the evolution of spatial reasoning during childhood is what can be leveraged in order to develop and acquire new knowledge, especially in the technology and science fields as Newcombe points out.



## 2.2.2 Tangibility and materiality

### *Tangible user interfaces*

In parallel to all the theories relating to spatial reasoning and cognitive development, there has been a lot of research trying to create learning experiences that enhance each one of the Piagetian development stages separately. But not much effort has been put in trying to combine them together for learning. Current approaches for teaching computational thinking and programming are online screen based. However, we are physical creatures. As Bret Victor mentions (Victor, 2011), 'We live in a three-dimensional world. Our hands are designed for moving and rotating objects in three dimensions, for picking up objects and placing them over, under, beside, and inside each other. No creature on earth has a dexterity that compares to ours'. Piaget also advocated for introducing physical interactions when designing educational methods. He highlighted the importance of physical actions towards objects in order to abstract from reality: "...there are individual actions such as throwing, pushing, touching, rubbing. It is these individual actions that give rise most of the time to abstraction from objects"<sup>12</sup>.

Following these same principles, Graspable User Interfaces were created. They are nowadays called Tangible User Interfaces (TUIs) and they are based on the manipulation of physical tokens in space in order to control a digital interface. In 1997 at MIT Media Lab, Hiroshi Ishii and Brygg Ullmer (Ishii & Ullmer, 1997) published an article where they talked about the connection between bits and atoms. This article focused on the junction between the cyberspace and the physical environment. Ishii and Ullmer pointed out that we had gone from using physical objects to measure events in the world, to using only computers and screens as unique measuring tools. In this article, Ishii and Ullmer tried to express their willingness to unify the richness of the physical world with the human-computer interaction. Tangible User Interfaces are a response to this need. One of the examples they presented in their research is MetaDesk. MetaDesk [Figure 12] is a platform developed at the Tangible Media group at MIT Media Lab in order to explore TUIs. The platform allows the use of 2D

---

<sup>12</sup> From Jean Piaget Quotes (2016). Retrieved from [jeanpiaget.org](http://jeanpiaget.org) (2018)

and 3D graphics on a screen, and the manipulation of various physical tokens detected by an array of optical, mechanical and electromagnetic sensors.

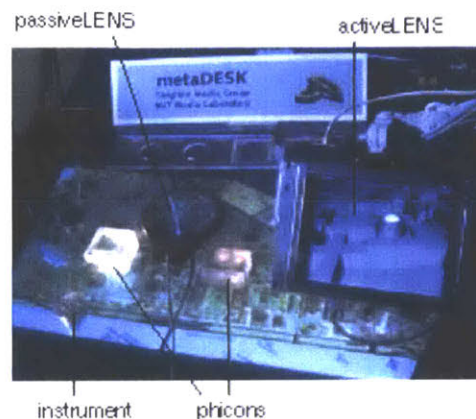


Figure 12. MetaDesk platform (Ishii & Ullmer, 1997)

Ishii and Ullmer's research triggered the development of a bunch of new projects and more research related to this area. One of the most successful projects using TUIs to the moment is Reactable (Jordà, Geiger, Alonso, & Kaltenbrunner, 2007), a musical edition platform.

Several research studies have claimed how Tangible User Interfaces are beneficial for the usability and natural interaction of a system (Schneider, Jermann, Zufferey, & Dillenbourg, 2011). Not only they improve the human-computer interaction methods but they also increase the learning outcomes if used in an educational setup (Skulmowski, Pradel, Kühnert, Brunnett, & Daniel Rey, 2016).

### ***Tangible programming***

Similar to how TUIs can increase the performance in learning environments, there is evidence on how tangible programming can help users in better understand their programming systems (McNerney, 2000).

The vast majority of platforms teaching computational thinking are online screen-based where the student gets disconnected from his own physical reality and isolated from others, sitting stationary looking at their own device. This is why, in parallel to all this movement of

online screen-based systems, researchers have been realizing the importance of embodiment and physical manipulation in learning experiences (Antle, 2009). Tangible programming draws from these cognitive theories and different tools have been developed following these lines of thought. AlgoBlock (Hideyuki & Kato, 1993), Tangible Programming Bricks (McNerney, 2000) or CyberPLAYce (Soleimani, Green, Herro, & Walker, 2016), amongst many others (Bers, 2010; Good et al., 2008; Peng, 2012; Weller, Yi-Luen Do & D. Gross, 2008), are some of the uncountable examples of programming platforms that make use of physical bits in order to teach about programming. If we take AlgoBlock as an example, the system allows students to plan and build procedures in group using physical blocks. The child does not program lines on code on a screen but instead, they use physical bits to design their programs and sequences. They can then observe the outcome of their actions on a screen.



Figure 13. Children using AlgoBlock system (Hideyuki & Kato, 1993)

A similar approach is used by systems like Tern (Horn & Jacob, 2007) or Quetzal (Horn, 2006), more recent programming languages that make use of the same principles [Figure 14]. They were both developed at Tufts University by Michael S. Horn and focus on the idea of making programming interfaces more accessible for the classroom, using simple materials such as paper to create programming inputs in space.



*Figure 14. Tem and Quetzal programming interfaces (Horn, Tangible programming with quetzal: Opportunities for education, 2006)*

The projects mentioned are good examples of tangible programming. The inputs of the system are laid in space making use of the physical environment of the user. The output is visualized on a 2D screen. This detaches the input actions of the user from the results that the user achieves. The tangible bits that the user can manipulate are physically disconnected from the digital output that happens in a different environment. In order to have an interactive and reflective system that naturally reacts to the user actions, the digital content should be attached to the physical tokens and react to those instantaneously so that the user understand their functionality automatically. Learning computational thinking with a tangible programming system can greatly benefit from overlaying digital content in space.

## **2.2.3 Augmented Reality**

### ***Augmented Reality in Education***

With the advent of Virtual and Augmented Reality (AR) and the latest advances in image processing and spatial tracking, these new technologies setup an ideal territory for educational experimentation. AR, in particular, lets us put in practice a lot of the concepts related to spatial cognition and cognitive development in the K-12 curriculum.

The affordances of AR allow us to take advantage of space and merge the virtual and physical worlds in a way that can greatly benefit educational and creative activities. Some researchers have already explored using AR as a means to improve learning and creativity approaches in education. However, the applications developed so far don't make use of the full potential of this technology. The new tracking systems such as ARCore (Google) or ARKit (Apple) and the new head-mounted displays such as Hololens (Microsoft) or Meta 2 (Meta) that have been released to the market in the last years open a new door to the exploration of new learning methodologies involving space and physical environment.

In different attempts to blend the digital on-screen based approaches with the tangible ones, some researchers have made use of AR as a flexible medium that can leverage spatial thinking capabilities in learning environments [Figure 15]. AR Scratch (Radu & MacIntyre, 2009) is a children's authoring environment for augmented-reality experiences that makes use of the Scratch programming language in AR. The system allows for 2D content in a screen, making use of image markers in front of the computer webcam.

A similar approach more focused on blending the digital with the physical environment is found in Color Code (Rosenbaum, 2010) by Eric Rosenbaum. Instead of image markers, Rosenbaum designed a set of experiments that overlaid digital content depending on color codes in the environment captured by the webcam.

Another example of the use of AR for education is Code Bits (Goyal, Vijay, Monga, & Pratul, 2016). Code Bits is an inexpensive tangible computational thinking toolkit that makes use of flat patterns and AR to teach programming skills. The user only needs paper to create image markers and a smartphone in order to visualize the AR content in space. In this case, the markers are flat and they need to remain in view in order for the application to display the

virtual content. The user needs to point at the board at all times in order for the digital content to appear. The application is not spatially aware.



*Figure 15. ARScratch (Radu & MacIntyre, 2009), Color Code (Rosenbaum, 2010) and CodeBits (Goyal, Vijay, Monga, & Pratul, 2016) Interfaces*

Finally, AvatAR (Fernandez-Baena, Adso, & Miralles, 2014) is an interaction system that makes use of tangible interaction and AR to control the character animation of a virtual avatar. In this case, a physical cube is used to blend between animations and control how the character is animated in 3D. The cube has to remain in view and the application is not spatially aware. A related system is used in SmartAvatars (Amores, Benavides, Comín, Fusté, Pla, & Miralles, 2012) where the user can interact with objects through virtual avatars in space. The application is not spatially aware and the object requires specific electronics in order to get connected to the smart device.

### **DynamicLand**

Another system that makes use of AR for computational thinking in learning environments is RealTalk (Victor, Schachman, Te, Horowitz, & Iannini, 2016). RealTalk was released in 2017 and it is “An environment for authoring and using computational media with ordinary physical objects — paper, pebbles, whatever’s at hand — recognized and brought to life by technology built into the ceiling”. The system was developed by a research group led by Bret Victor. Later on they created a whole environment on top of RealTalk, called DynamicLand (Victor B. , DynamicLand, 2017), a space for collaboration, programming and tinkering. Bret described DynamicLand as “a medium that we can see, feel and manipulate”. DynamicLand is a space where people can interact with objects and the physical tokens get merged with the digital content projected from technology installed on the ceiling [Figure 16].

The 2D digital content gets projected onto the surface of a table where the user can interact with it using any type of physical object. The interaction is bound to the table where the technology is set up.

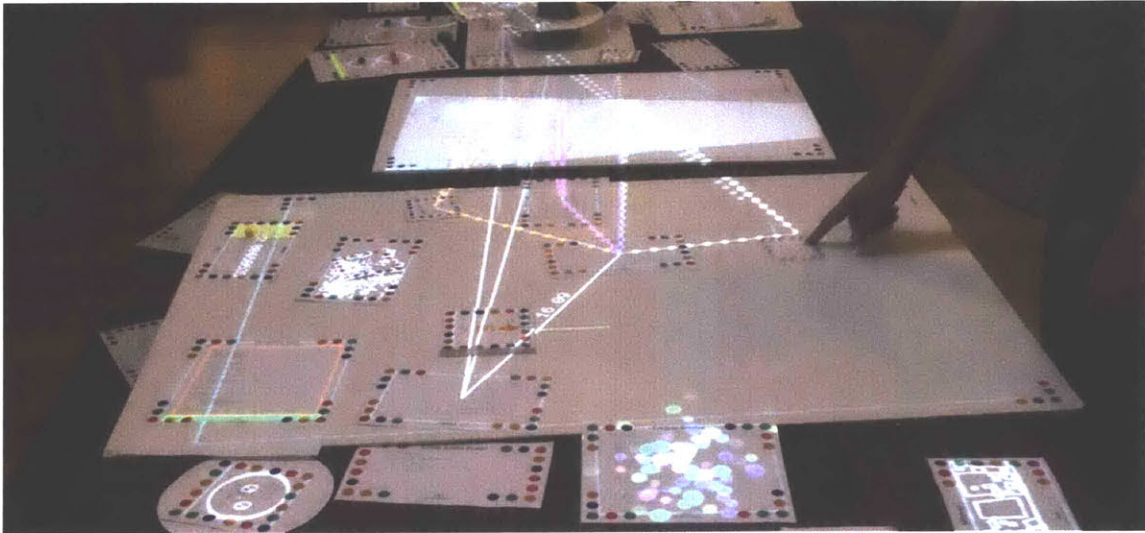


Figure 16. *DynamicLand* (Victor, 2017)

### ***Spatial Object Programming***

AR has been used also as an interface to manipulate our physical environment. The visual component creates a direct link between the digital content and the physical objects linked to it. This is what Valentin Heun explores in his research and the development of his AR system, the Reality Editor (Heun, Hobin, & Maes, 2013). In Heun's words, "If an object can become digitally linked with a virtual object, one can form methods for an expressive form of programming that provides a new form of control over physical space. Such a programming language needs to be easily accessible and operable so that everyone can adjust the world around them".

Heun presented what he called Smarter Objects, which could be programmed with the Reality Editor. The Reality Editor is an AR interface to create programs and manipulate the physical environment around the user [Figure 17]. Heun's goal is "...to create physical objects that have a virtual component to benefit from their physical design while making them

adaptable and extensible". He thought about the AR interface as a visual elongation of the physical qualities of the objects in the environment.

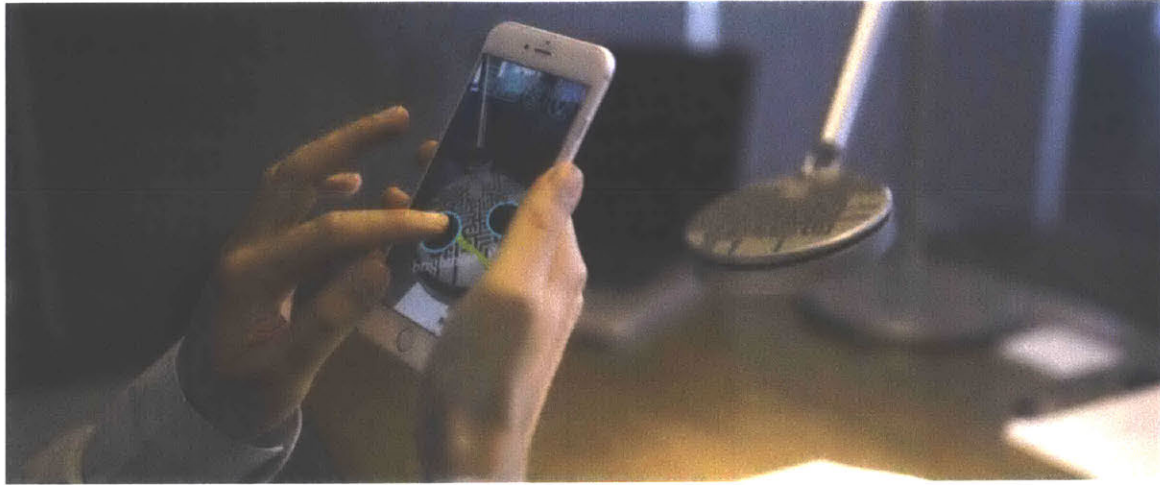


Figure 17. *Reality Editor* (Heun, Hobin, & Maes, *Reality editor: programming smarter objects*, 2013)

### **Augmented Creativity**

Following the principles that define the Lifelong Kindergarten group at the MIT Media Lab, other researchers have used AR to boost creativity and play in an AR environment. Zünd, Fabio, et al. (Zünd, et al., 2015) introduced the concept of Augmented Creativity as a way of bridging the real and virtual worlds to enhance creative play. They present different applications that make use of AR to boost the creative abilities of children. In one of their examples a system is developed to enhance robot programming using AR [Figure 18] (Magenat, Morderchai, Klinger, & Sumner, 2015).



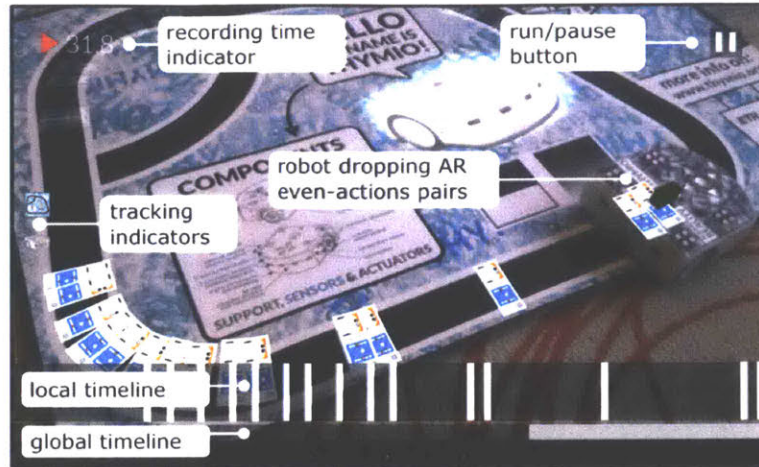


Figure 18. Enhancing Robot Programming with Visual Feedback and Augmented Reality (Magnenat, Morderchai, Klinger, & Sumner, 2015)

AR is setting the stage for a new era in the educational landscape. The use of spatial awareness and physicality in the digital medium allows for a vast amount of new interactions that remain still to be explored.

### 3 Embodied Spatial Programming

#### 3.1 Humans and our spatial mental models

We have talked about theories of spatial cognitive development and how different approaches see the evolution of spatial domains in the brain as an earlier or later event in childhood. Yet, all of the cited research concurs that spatial models created in our brain from early stages can be leveraged to enhance other human capabilities linked directly to those brain areas.

We humans, as a physical entity, live and exist in a four dimensional world. At every moment in time we exist at a position in space and we relate to objects and other entities around us that exist at another position in space. Everything we do happens in space. Even our dreams take place within locations with spatial impressions, though they are abstract and imaginary. We relate all of our actions to space since the moment we are born. This leads us to build a set of spatial models in our brain that govern our way of interacting with things almost in an unconscious manner. When we walk up the stairs, we do not stop to calculate how far we have to move our leg in order to get to the next step; our leg moves the appropriate distance because we have a mental model in our brain that has evolved from necessity and that has permeated in our brain until it is part of it.

Human evolution has shown us that we have benefited from these base spatial models to build improved interfaces. Let's take an early example, so ancient that its exact origin is still unknown; the abacus. The abacus is a tool developed for practical mathematical calculations.

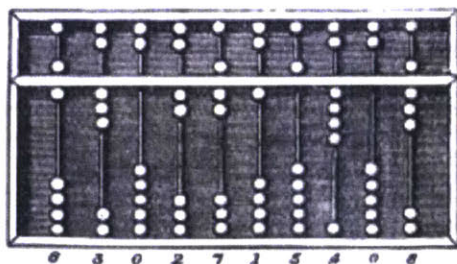


Figure 19. Chinese Abacus<sup>13</sup>

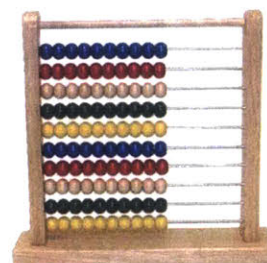


Figure 20. Modern Abacus (20<sup>th</sup> Century)<sup>14</sup>

<sup>13</sup> From Wikipedia (2018). Retrieved from [en.wikipedia.org/wiki/Abacus](https://en.wikipedia.org/wiki/Abacus)

<sup>14</sup> From Flaghouse (2018). Retrieved from [flaghouse.com](https://flaghouse.com)

The abacus has a physical and spatial interface. Slide right to add, slide left to subtract in powers of 10 (in more modern versions of the tool). Adding and subtracting are two mathematical constructs that humans found to be abstract and difficult to escalate without a physical concrete representation. The solution to learning these difficult mathematical constructs ended up being an interface that made use of spatial models in our brain. We found a way of using our spatial notion of an object position to represent and operate with mathematical abstraction.

We have used these techniques for years and nowadays rely on them even more. One quintessential example of this is the Graphical User Interface. Graphical User Interfaces were introduced as a way of simplifying the steep learning curve of command-line interfaces. In 1945, Vannevar Bush published an essay that he titled “As we may think”, talking about the problem of the current selection and database systems. He highlighted the fact that the human mind works according to association not according to alphabetical or numerical systems. This early essay inspired the creation of SketchPad (Sutherland, 1964), the first graphical computer-aided design program, created by Ivan Sutherland.

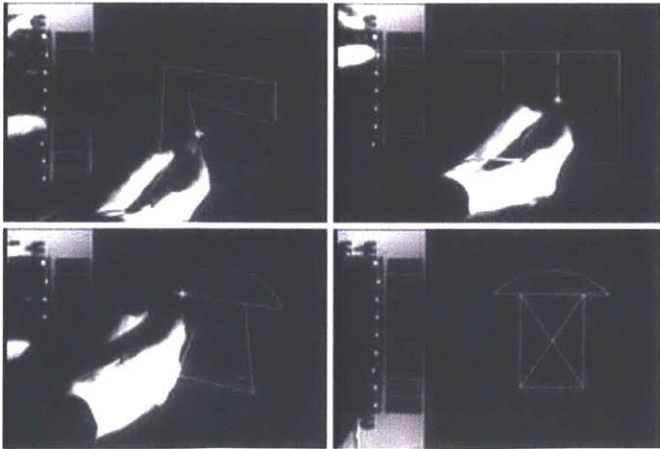


Figure 21. Ivan E. Sutherland demonstrating Sketchpad<sup>15</sup>

<sup>15</sup> Inernet Archive (Kay, 1987). Retrieved from Wikipedia (2018).

*SketchPad* [Figure 21] made use of spatial mental models to design geometry in a 2D interface. This early vision of a computer interface inspired the creation of the On-Line System (NLS) (Englebart, 1986) in the late 60s by Douglas Engelbart, who introduced the *mouse* as a new input device to control these 2D interfaces. Similar to the abacus, the mouse allows the user to move their hand along a plane in space to control an information system. It is another example of how we have benefited from our spatial models to control a non-tangible information system.

Later on, Alan Kay developed a new GUI for the Xerox Alto computer [Figure 22] that was based on windows, folders, containers and bins placed in space and controlled by the *mouse*, the first computer “desktop”.

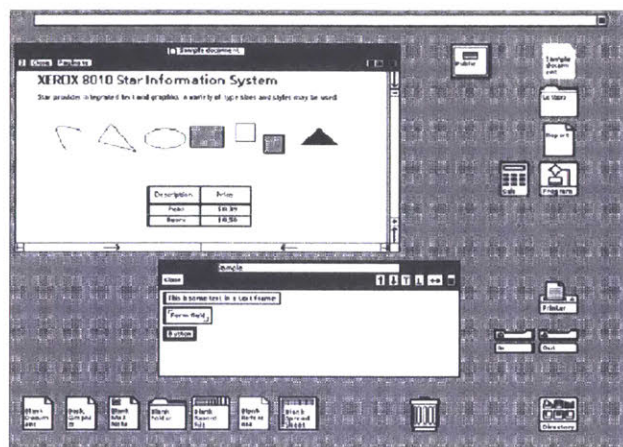


Figure 22. The first Graphical User Interface on the Xerox Star 8010 (Bewley, Roberts, Schroit, & Verplank, 1983)

This new Graphical User Interface placed elements in different positions in the 2D plane and made use of *skeuomorphism* to make the items represented resemble their real-world counterparts. As well as making use of visual mental models that the user already had, it made use of spatial mental models in order to manipulate information in the system.

Our interfaces started being three dimensional; like the abacus. We used 3D objects that we could manipulate in space. Yet, with the computing era our main interfaces shifted towards 2D screens anchored in a single point in space where the main computing unit was. The first

computers wouldn't allow for the use of space as their mobility was limited. With time, screens have pervaded and the interactions with the systems have been reduced to a tap of a finger. Mobile devices have overrun the social scene but, we are still docked to the flat 2D interactions that don't make use of space or the physicality of our environment [Figure 23].



*Figure 23. Wall-E movie depicting a future with inactive humans surrounded by 2D screens<sup>16</sup>*

Concurrently, other types of interactions have been developed taking advantage of the sensory interfaces of our body. Voice and speech recognition or brain computer interfaces are technologies that have become a big spotlight in research and start to enter the commercial landscape.

All these input/output systems that we are so eager to take further technologically, could gain great value by recovering the spatial component that our original interfaces were built upon. We can, for example, have a kid in a room listening to a story by a voice assistant. But what if the voice assistant was spatially aware and could tell the story making the kid move to the different rooms in the house, battling dragons on top of the sofa, or hiding from them behind a wall? We would not only be enhancing her intellectual skills and attitudes but also improving motor skills and linking them to a dimensional component directly connected to her physical surroundings.

---

<sup>16</sup> From Strauss, Mark (2014). Retrieved from Gizmodo.com

We can teach the kid how gravity works by showing her on a screen or we can track a physical ball falling and project the acceleration values on top of it.

We are squashing our interactions into a 2D flat world with touch, or a one channel dialogue with voice commands where we don't have to move, we don't make use of the full capabilities and the sensing inputs and outputs of our physical being.

The three dimensional nature is what makes us humans, what lets us grasp things, what defines us in the world. As mentioned before, our brains evolve with spatial models that consider the dimensionality of our bodies. Interacting with systems that use flat surfaces, voice interfaces or brain interfaces require new mental models that we have to infuse in our minds. We are currently creating these models without building upon the foundations of our intellect. That is the reason why seniors have a hard time understanding how to use a touch screen. Because it is an interaction model that does not match any of their mental schemes. Children are able to adjust faster to these new interfaces as their mental models have not evolved and permeated so much. Nonetheless, by offering them flat interfaces we inflict in them an interaction frame that restricts their contact with their surroundings. In doing this, they lose part of the proficiency of their limbs interacting physically with the environment [Figure 24]. From using their whole hands to grab a toy and move it in space, we offer them an interaction with just the tip of their finger to play Candy Crush Saga in an 8-inch screen [Figure 25]. Deft manipulation of objects and assimilation of spatial awareness get lost with these types of interactions.



Figure 24. Child playing with well pump. Ten fingers, both hands.<sup>17</sup>

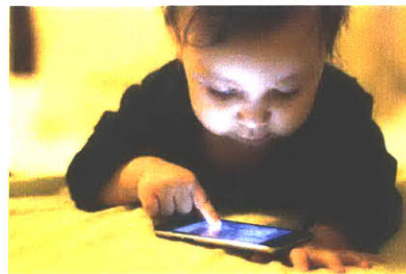


Figure 25. Child playing with touchscreen. One finger, one hand<sup>18</sup>

---

<sup>17</sup> Retrieved from Sciencing.com (2018)

<sup>18</sup> Retrieved from flickr.com/photos/courosa/ (2018)

The flexibility of the digital medium has opened the door to new interaction models and schemes that we can take advantage of. Education, arts, sciences, health are just some of the areas that have greatly benefited from new interaction models that the digital medium provides. The current visual models we live with have cemented interaction practices mainly rooted in screens; we have to work towards a future where we don't lose contact with the spatial and physical models.

The interaction models we are shifting to don't make use of the spatial domains that our mind masters during childhood. This doesn't mean that we shouldn't be using these interaction models but we should find a way of leveraging these spatial domains and integrating them in the new digital arena.

## 3.2 Augmented reality, the body and space

### 3.2.1 Do not gesture nonsense in space

One of the technologies that has great potential for leveraging these spatial and physical mental models is Augmented Reality. There is still a lot to be done, however, with all the budget being spent in AR software and hardware, one can easily *future-cast* how quickly the devices being developed are going to become more usable and wearable.

Hololens (Microsoft), Meta 2 (Meta), Magic Leap One (Magic Leap) or Project North Star (Leap Motion) are examples of AR headsets that battle to win the masses: some of them, untethered; some of them, lighter; some of them with relatively wide fields of view. They are still considerably expensive and uncomfortable if we dare to compare them to our day to day wearable devices such as, for example, a watch. One of the essential characteristic that gives value to a wearable device is the ability to become ubiquitous to the user. I would consider that a wearable device is effective if the user is capable of acting naturally, proceeding with certain tasks and forgetting that she is wearing it. The device needs to do a good job at extending human capabilities without disrupting the person's natural actions. We don't think that we have a hand when opening a door. We move the hand towards the handle almost instinctively. A wearable device should be as comfortable and as ubiquitous as having a hand. Unfortunately, the existing AR devices [Figure 26] are far from being comfortable. No one would imagine themselves walking around with these, as if nothing was happening:



Figure 26. Hololens, Meta and Magic Leap One Head Mounted Displays



Nevertheless, the revenue for AR devices and content is hitting unexpected highs. And so all these leading companies are willing to spend more budget, mainly on hardware, to have better devices. So it is not hard to imagine what will happen with AR devices in a very Moore's Law style. In a few years, these devices will probably look more like this:



*Figure 27. Regular glasses*

They will look and feel like regular glasses [Figure 27]. Or like a regular eye lens. And then they will be comfortable and they will be ubiquitous and they will be effective to enhance our human capabilities in a way that they don't disrupt others. It is in this future setting, where digital content will be displayed in front of our eyes, that we have to consider the interactions that we will be able to perform with this digital content.

All of the AR devices being developed are taking into account our hands and our arms as root pinpoints for the interaction with the digital content. And most of them perform hand tracking, allowing for a direct manipulation of the virtual elements with our hands in space. So a common image we have in mind when thinking about these devices is this one:



*Figure 28. Meron Gribetz interacting with Meta headset*

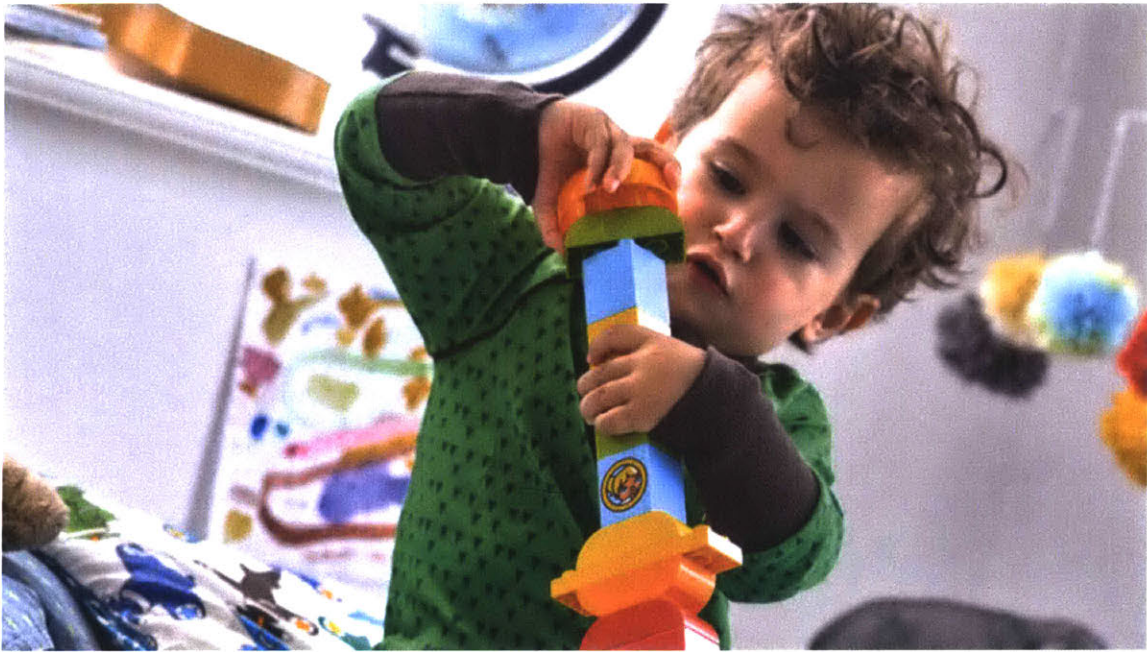
This is Meron Gribetz [Figure 28], founder of Meta, a company that makes and sells AR products. He is wearing the Meta headset and raising his hand to interact with some digital content that is displayed in his headset. He may be interacting with a drop down menu or petting a kitten. We will never know.



*Figure 29. Hololens user interacting with Hololens system*

This is a Hololens user [Figure 29] wearing a Hololens headset. He is also interacting with digital content. He may be holding a *Prophecy Orb* or trying to go to the home menu. And again, we will never know.

All these gestures that the user has to perform in front of their faces to reach to the digital content are not natural. We don't go about our lives moving our hands aimlessly in front of our face to achieve a goal. We interact with our environment, we touch, we grasp, we press, we manipulate the physical objects that we have around us. That is what our body is used to as an entity that exists in a physical world and in a physical space. These are the gestures that we should leverage when designing new spatial interfaces for digital literacy [Figure 30]:



*Figure 30. Kid playing with blocks*

### 3.2.2 Towards a graspable future

The potential of AR lays in the spatial awareness and the capability of spreading and arranging the digital medium on our immediate physical environment. New machine learning techniques have opened a door to a new landscape for image processing, space tracking and object recognition. We are now able to know the exact position of an object in space and we may even be able to connect to it if it has smart object capabilities. We can track walls and obstacles in order to cover digital elements faking occlusion. We are basically able to extract the digital medium from the flat screen, place it everywhere on the environment and link it to physical tokens. And one of the most important factors, anyone with a smartphone can interact with it. AR is becoming an accessible medium that the masses have access to and are willing to use. Still, the most notorious application developed with it is a game with creatures appearing randomly in space that you have to catch by swiping your finger on the screen (Pokémon GO) or an application of a dancing hot dog (Snapchat).

The Augmented Reality medium is the perfect arena for applying the spatial and physical constructs that we have evolved with. We can now endow the objects around us with a digital extension in order to understand their properties or in order to interact with them in new improved ways. The digital medium should be inherent to the physical space in a systemic way.

Let's use the gravity example: if we want to teach some basic physics to a child, we can design a whole AR application with fake digital balls that fall in front of the user and respond to a physics engine. We could also show the same content on a 2D screen. So why would we bother doing it in AR if we are teaching the same concepts in the same way? On the other hand, we could have a physical ball that we track and we could display mass values, acceleration values and so on. The child could hold the ball, as she has always hold a ball and played with it [Figure 31] but now, she could be able to understand what is going on when throwing it up on the air. We could even display elasticity values, or even energy retained by the ball when squashing it. She could feel the mass of the ball and experience the event at true scale with all the fidelity of their combined senses.



*Figure 31. Gestures we are used to perform*

This is a tricky example as physics is an inherent characteristic of the physical medium. Let's think about a more abstract example: someone reading a book or writing a document. We could have an AR interface with a digital book. The user swipes pages gesturing in space, in a very 'petting kitten' style, like Gribetz did. If we stop to think about it, probably most of you would still prefer to read this book on the original version, paperback being able to turn pages physically [Figure 32], or even on a Kindle. It is not very natural to turn pages swiping your arm in the air or type on an imaginary typewriter floating in front of you. What if we had this physical book but could extend its contents adding a layer of digital medium on top of it. We could still turn its pages but we could also look for references, see visuals of the different sections and words tapping on them, and so on. An example of an augmented book is *Variable Reality* (Yang, 2015). The digital medium anchored to the physical token relates directly to the constructs deep-rooted in our mental schemes.



*Figure 32. Turning a page of a book*

We have the capabilities of making AR graspable, of relating it to the physical environment that we are used to interact with. This is the way in which we create meaningful interactions. This is the way in which we shorten the learning curve of a new digital interface.

### 3.3 Authoring to own the medium

AR creates a new landscape that allows for new interactions and new ways of exploring the digital medium in space. AR becomes a new medium per se and we need creation tools native to this interface in order to design and develop content and interactions for it.

HyperCard, LOGO, Scratch are examples of tools to allow authorship in a way that's native to the computing medium. Painting and sculpting constitute authorship for Fine Arts, the same as playing the guitar or singing conceive authorship for music, writing gives authorship to literature or coding gives authorship to computing. All of these happen at different levels and offer possibilities to expand on each one of the mediums. The same happens with Augmented Reality. We need authorship tools native to this spatial digital framework to understand, master and open new spaces in it.

AR is a technology that was developed years ago. In 1968, Ivan Sutherland created what he called The Sword of Damocles, considered to be the first Virtual Reality head-mounted display and often referred to as an early precursor of AR (for its translucent display and other characteristics) [Figure 33]. It has not been until recent years that AR has become a pervasive medium accessible to the masses and capable of being completely spatial and physically aware.

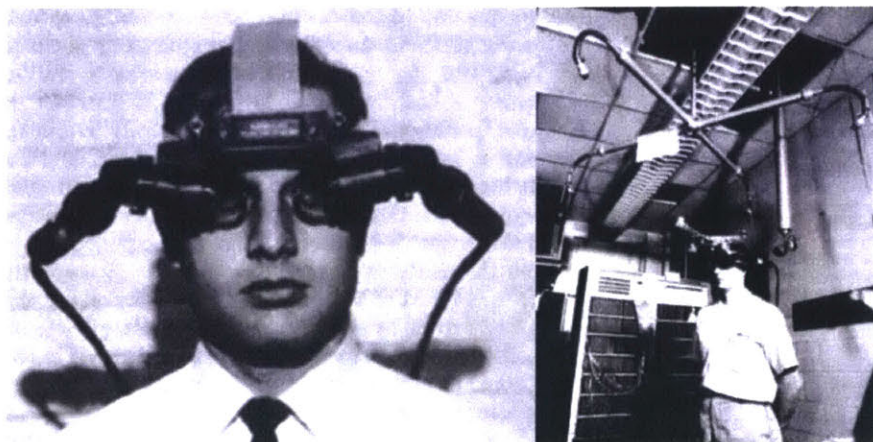


Figure 33. *The Sword of Damocles, 1968*<sup>19</sup>

---

<sup>19</sup> Retrieved from Sutori.com (2018)

When a new technology is developed, it is usually created with a specific means; to solve a problem or to proceed with a task that was not feasible before. Nonetheless, by trying to solve a problem we often allow, without being aware, for new forms of expression that can be exploited and taken further. That is the main reason why it is important to build upon these new platforms in a way that we make them accessible and malleable by creators that can empower new uses for them.

These creator tools for AR should follow the good practices and principles we have been exploring, such as binding the content to spatial constraints or anchoring it to physical tokens. Moreover, they should take advantage of the characteristics of the medium to teach digital literacy in a way that is more ingrained and easy to digest by the user.

These are the principles that an AR creation tool should follow:

- Leverage spatial mental models: design interactions that draw from spatial models that the user is accustomed to.
- Track surfaces and consider the position of the user relative to them when anchoring content to space.
- Use physical interactions: bind digital content to dynamic physical tokens in order to make the interaction more intuitive and inherent to the human physical quality.
- Avoid placing content floating in space without a reason.
- Take advantage of the flexibility of the digital medium and create reflective and interactive tools that evolve in real-time with the user.

These principles constitute the optimal ground for a medium that is well-suited to teach digital literacy. Entrenched in our surroundings, it constitutes a bridge between the physical constructs that we already have assimilated and the digital tapestry that keeps on growing in our society. This are the principles that define the concept of Embodied Spatial Programming.



### **3.4 Embodied spatial programming for computational thinking**

Using space as the understructure and the mental spatial constructs as the building blocks, we can design a playground for digital literacy that acts, also, as an authoring platform. This digital medium conformed by the computing layer offers a perfect setting for learning computational thinking concepts while using the medium to author in space.

To that end, we propose the use of Embodied Spatial Programming as a programming paradigm that allows us to create using the dimensionality of the environment. Embodied Spatial Programming makes use of space and the physical objects around the user in order to design new behaviors and programs that have a direct effect on the physical surroundings of the person. The user is able to code through tangible objects and mechanisms; the result will be immediately displayed through digital content overlaid on top of the physical environment of the user as an instantaneous response from their actions. AR's capability to bridge the physical and virtual worlds make it a suitable technology to apply Embodied Spatial Programming principles to the interaction design on new educational interfaces and applications.

When programming, we make use of complex toolsets and visualize the final result in other devices such as screens, or physical mechanisms such as robots. It is a hard exercise to abstract from the input medium and try to program in lines of text that will later become actions. It is the theory of Kay reversed in a way where instead of going from "Doing with Images" to "Symbols", we have to configure this symbols to get to the desired images. These different outputs have some advantages and disadvantages. For example, as we have already mentioned, the screen's limitation is that it is a two-dimensional interface, a flattened simulation of our own reality. Nonetheless, it is capable of displaying a wide variety of content ingrained in the digital medium.

On the other hand, when we program physical mechanisms, such as the LEGO Mindstorms (Klassner & Anderson, 2003), we can take into account the physical environment and code for a three dimensional output but the interactions and possibilities get narrowed down to a few commands constrained by their physical affordances. The physical mechanisms don't have the flexibility that the digital content can have and the screen offers a simulation on a two-dimensional environment detached from our surroundings.

With engaging in Embodied Spatial Programming, we make use of tangible programming but the result gets displayed on top of our physical environment, with AR, making use of the flexibility of the digital content laid in space and embedded in the user's surroundings.

Programming involves understanding and making use of an abstract language in order to generate an output on our physical world. This output can change the behavior of a robot, change the representation on a screen or configure our thermostat to turn on when we are ten minutes away from home. When we code, we make use of this language by generating scripts and text files that act as the input; they get interpreted and compiled to generate an output, often on a separate system, this being a screen, a robot or other device.

With Embodied Spatial Programming, this input and output are merged in the same environment [Figure 34]; the coding process happens in space, overlaid on top of the direct output of the system. The programming language is embedded in space. The user immediately understands what that code is generating or modifying. We make use of the physical tokens as our input and display the digital medium on top as the output of this arrangement in space.

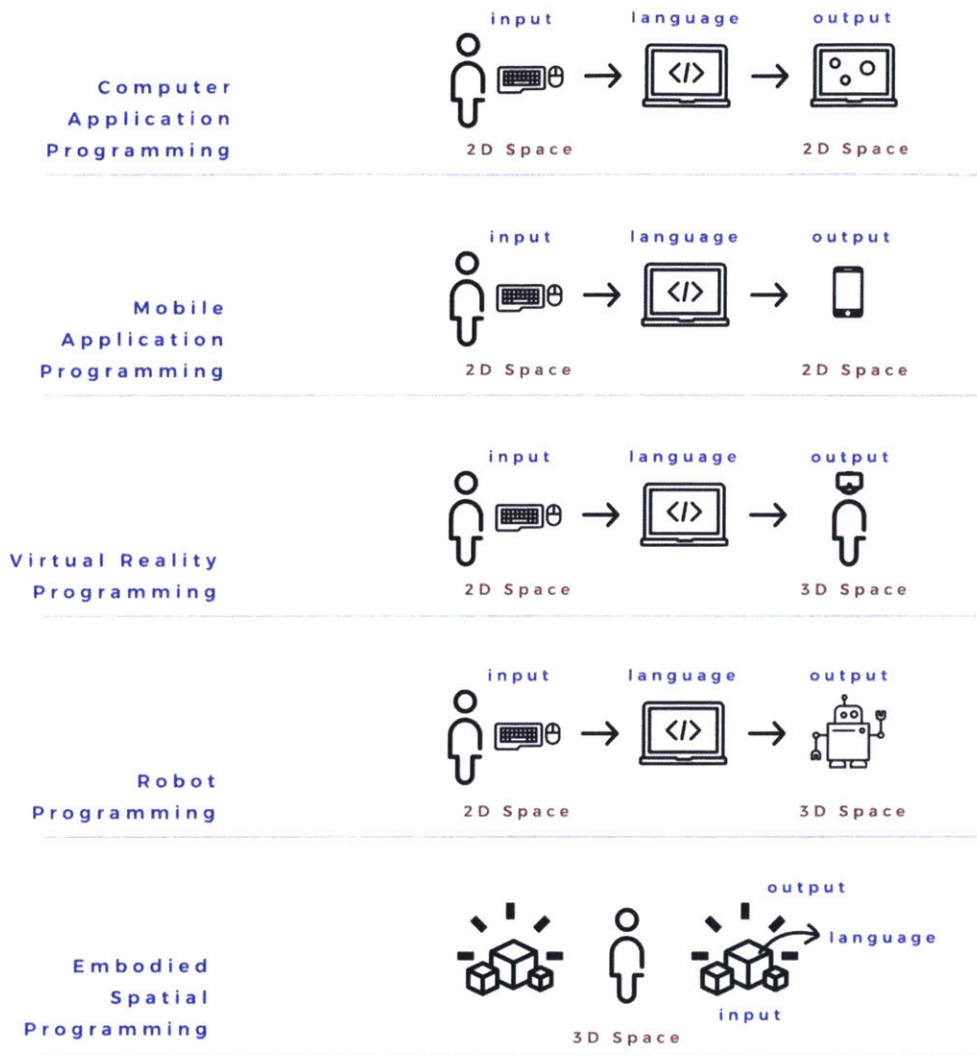


Figure 34. Programming examples vs. Embodied Spatial Programming

Different authoring languages and tools have made use of 3D space as the output medium when programming. The Logo Turtle [Figure 35] is a clear example. The user programs on a 2D environment, a screen, and the output is experienced in 3D, where the turtle draws in space.



*Figure 35. The Logo Turtle (Papert, 1980)*

Embodied Spatial Programming makes use of space, not only on the output but also on the input. The user defines the behaviors in the 3D environment and experiences the results immediately and directly linked to her actions on the 3D space.

The idea is having a programming paradigm that adapts to the user nature instead of having the user adapt to a complex system represented in a completely detached environment. This makes Embodied Spatial Programming a perfect archetype for teaching computational thinking concepts and coding principles.

## 4 HYPERCUBES

### 4.1 Overview

#### 4.1.1 HyperCubes outline

Following the Embodied Spatial Programming principles mentioned above, we have developed HyperCubes, an Augmented Reality application to teach Computational Thinking in space. HyperCubes is a practical example of how to make use of AR to build an Embodied Spatial Programming platform for teaching computational thinking. We can define HyperCubes with the following words:

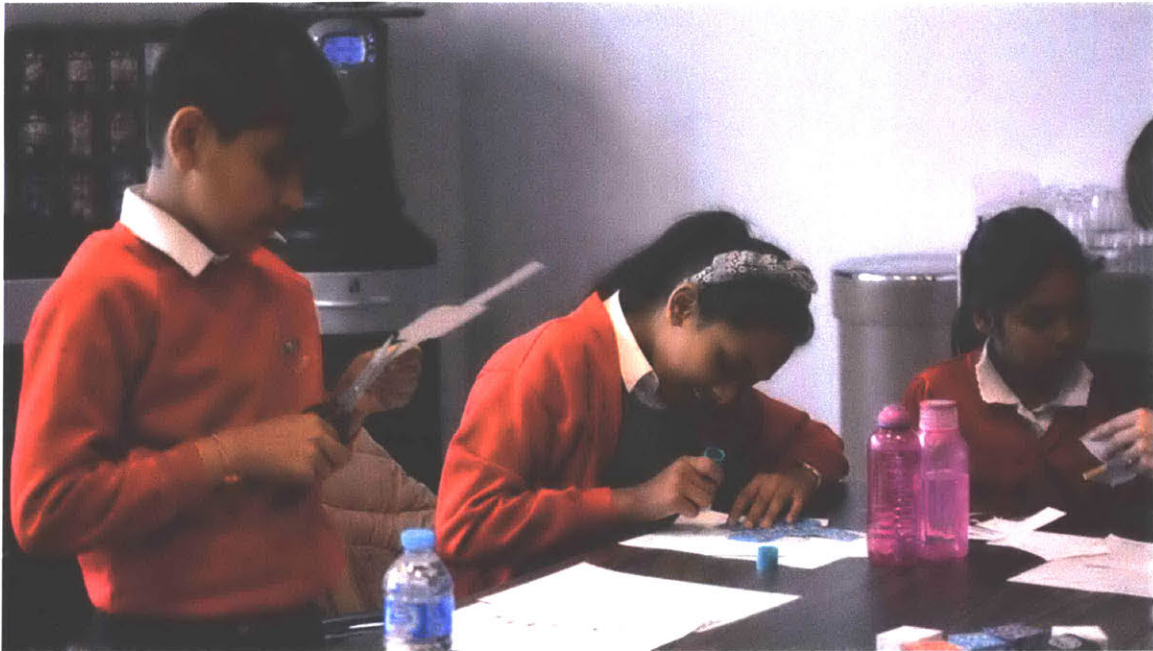
**HyperCubes is a DIY Augmented Reality platform to learn Computational Thinking making use of a set of paper cubes to create and control digital content in space.**

Each cube has a different set of behaviors inherently attached to it that relate to basic concepts in the K-12 curriculum. The AR application detects the cubes and displays a flow of objects in space that the user can control by placing the cubes in different configurations. Each cube is related to a different programming construct or concept and has a different behavior inherently attached on each one of its sides. The user, by tracking the cubes with an AR mobile application, can create virtual objects that flow in space in a certain direction. These virtual objects can have different nature; they can be pure geometry such as a sphere or a cube, or they can have the form of more specific elements such as animals, rockets or a person.

#### 4.1.2 DIY culture and maker culture

HyperCubes has been designed to promote the DIY movement and culture, where the user is considered as a self-sufficient entity capable of making their own tools and every-day objects. Children are increasingly involved in the DIY culture involving digital media and new

technologies (Kafai & Peppler, 2011). Following the arts and crafts foundations to encourage handmade work, HyperCubes is thought to give the user the opportunity of making their own cubes with material as simple and accessible as paper. The Maker culture sets a framework for HyperCubes to boost the concept of learning-by-doing and tinkering with physical objects rather than limiting it to a screen with digital content.



*Figure 36. Children making markers for HyperCubes (Leicester Art-AI Festival 2018)*

The child builds the cubes with paper, scissors and glue. The templates for each cube have to be previously printed with a regular printer. Once the cubes are mounted, the child can use them as markers for the AR application. This gives the child a feeling of ownership as it is something they have built with their own hands. It is also a physical token that they will keep once they have finished using the app and that can be reused later on during a new session.

### 4.1.3 Tangibility and raw materials

As mentioned, the markers for the application are made using raw materials such as paper. The children just need paper, scissors and glue in order to create the cubes. This makes the application accessible to anyone without the need of having extra electronics or any other complex components.

Using physical paper as the basis for the interaction is beneficial for the learning experience. The student has a connection with the physical environment and it is an object made of a very familiar material.



*Figure 37. Paper cubes made by 10-year-old students (Leicester Art-AI Festival)*

### 4.1.4 Mobile device as an accessible medium

One of the strong factors of the HyperCubes platform is the fact that it is designed to be accessible to anyone who has paper, scissors, glue and a mobile device. It does not require expensive technology such as head-mounted displays or specific hardware for the cubes to work. This makes the platform easily deployable in any kind of environment being it a school or at home.

### **4.1.5 Embodied Spatial Programming in AR**

The vast majority of the current systems and platforms to teach about programming provide learning environments for computational thinking; however, none of them make complete use of the 3D space in their experience. Some of the systems use 2D projections and others make use of 2D content on the 3D environment. Our approach intends to make full use of space and the physicality of the environment in order to enhance the learning and creation process. We want to design an authoring platform that follows the principles of Embodied Spatial Programming:

- Leverage spatial mental models
- Track surfaces and consider the position of the user relative to them when anchoring content to space.
- Use physical interactions
- Avoid placing content floating in space without a reason.
- Take advantage of the flexibility of the digital medium and create reflective and interactive tools that evolve in real-time with the user.

The main idea in HyperCubes is to give the child a framework to play and learn from basic programming constructs linked to physical tokens in space.

### **4.1.6 Paper Cubes and HyperCubes**

HyperCubes is based on a first prototype developed as an AR experiment called Paper Cubes (Fusté, Amores, & GoogleCreativeLab, 2017). Paper Cubes [Figure 38] consists of an AR application where the user can control the behavior of some stick figures by placing paper cubes in space. In Paper Cubes, there is a limited number of cubes and only one behavior inherently defined on each cube. It is an application that makes use of Embodied Spatial Programming to define a playful experience. HyperCubes goes beyond this idea and allows the user to specify variables and content on the cubes dynamically through AR. For example, the user is able to change variables assigned to any cube at all times, customizing the sequence of code that gets generated. HyperCubes is also spatially aware and allows the



user to move freely. The content gets anchored in space and it is persistent when the user moves away from it. This gives the user much more freedom and enhances the creativity and learning processes.



Figure 38. Paper Cubes Google AR Experiment (Fusté, Amores, & GoogleCreativeLab, 2017)

## 4.2 Application design

### 4.2.1 Learning computational thinking with HyperCubes

#### *The set of cubes*

The application is based on 8 types of cubes with different properties. These 8 types of cubes are the following:

- **Emitter Cube:** Creation of digital objects
- **Transform Cube:** Transformation of digital objects
- **Split Cube:** Instancing digital objects and creating paths
- **Logic Cube:** Logic gates for asynchronous flow of digital objects
- **Physics Cube:** Physics simulations applied to digital objects
- **Effects Cube:** Visual effects and animations applied to digital objects
- **Gaming Cube:** Gaming elements to create games with digital objects
- **Hyper Cube:** Saving and loading virtual scenes previously generated

Each one of the cubes has 6 different behaviors; one behavior per side of the cube, except for the Hyper Cube which has only one behavior in total. Each behavior may have different parameters that can be tweaked to adjust its functionality, such as a parameter to control the speed of the virtual objects or a parameter to select the color of the objects. Each type of cube can be printed, mounted and used multiple times on a scene.

The user lays the cubes on a surface and tracks them with the AR application. Each behavior will be activated when the application tracks the cube and identifies the behavior as the one on top. If the user wants to change the behavior she will have to rotate the cube in order to have the desired behavior on the top side of the cube.

Once a cube is tracked, a visual feedback will appear around the cube to let the user know that the cube was identified. Once recognized, the cube will be anchored in space. If the cube is no longer in view, the application will remember the position of the cube in space. Once tracked, the cube doesn't have to be in view in order for the user to visualize the digital content.

When the user taps on the cube, a graphical user interface appears with a side menu. The user can tweak the parameters of each behavior by interacting with this interface [Figure 39]. If the user wants to delete or replace a cube, she will have to remove the old cube using the GUI on the application. Then she can proceed to track the new cube.

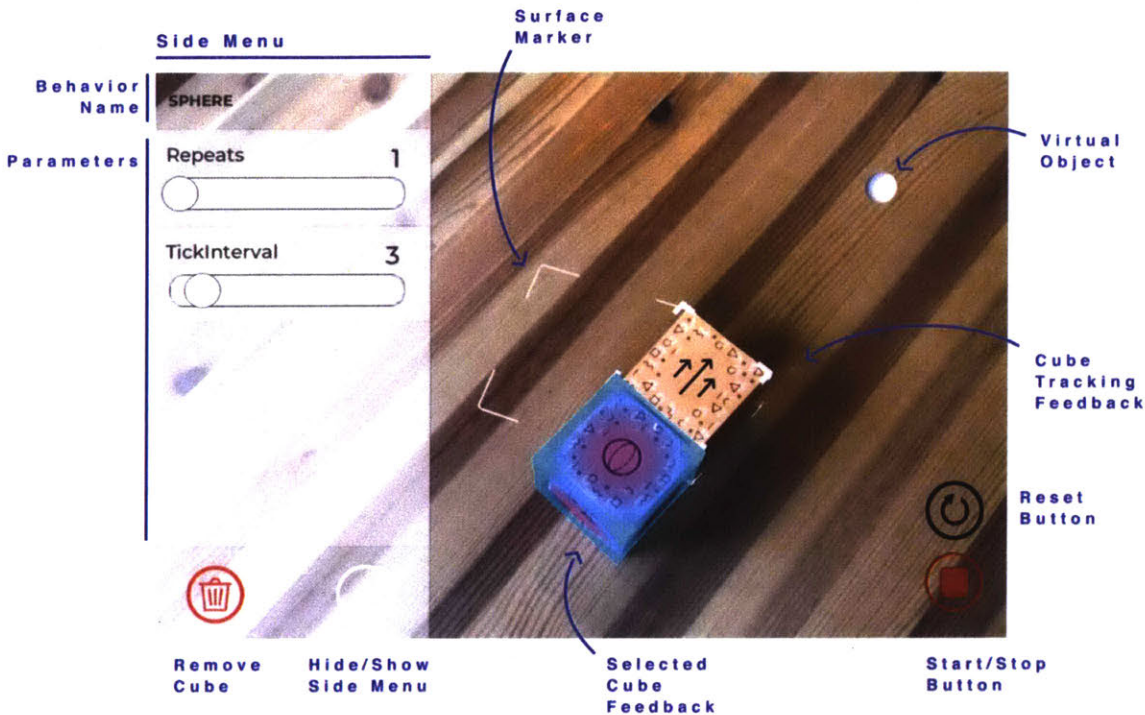


Figure 39. HyperCubes Application Interface

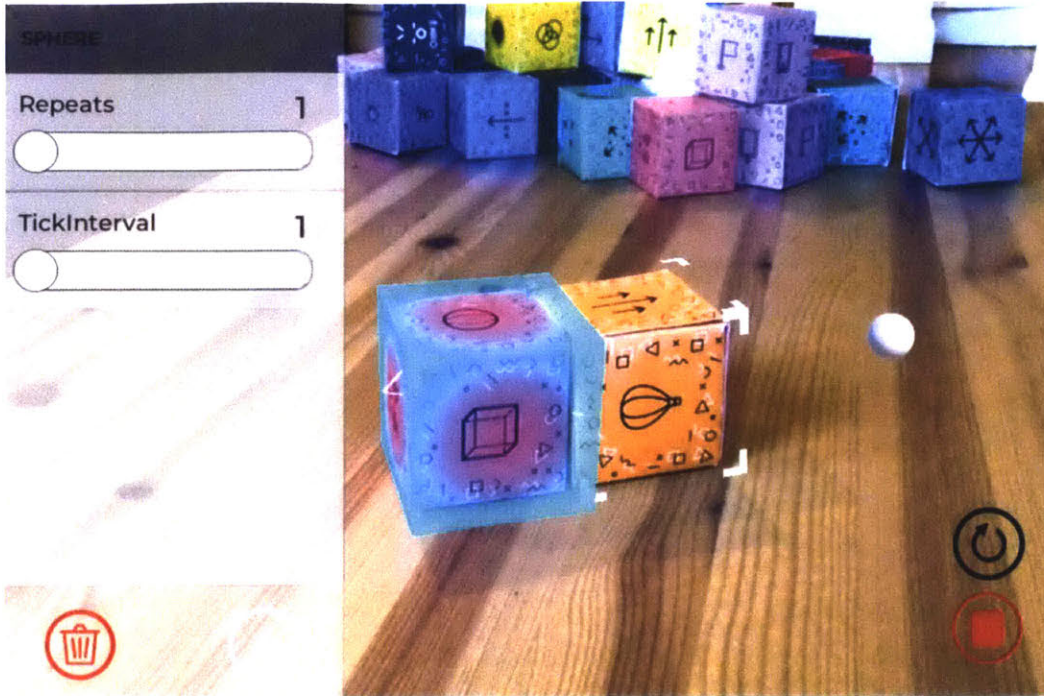


Figure 40. Side view of HyperCubes Application Interface

### **Emitter cube**








Figure 41. Emitter cube

The Emitter Cube [Figure 41] works as a cube that creates the digital content. It is a memory allocation simile and it generates geometry in different shapes. The geometry object is the base unit of the application. It flows in space and its behavior is constrained by the rules of the cubes that are laid in its path. The emitter cube is the starting point of the application. Without an Emitter cube there is no content in space as there is no other cube with creation

capabilities. The function of the Emitter cube is to generate the virtual objects. These objects are generated inside of the cube and won't be visible until another cube that makes them move, such as the Transform cube, is placed next to it.

The Emitter cube has 6 different types of virtual objects that can be generated. Each type is associated with one of the sides of the cube. All of them have two common parameters: the *Repeats* parameter and the *Tick Interval* parameter. The *Tick Interval* parameter controls how often the boids are generated. The *Repeats* parameter controls how many times the Emitter will emit a boid. It works as a loop in case the user sets the *Repeats* to more than one.

The different sides of the Emitter cube are the following [Table 1]:

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	SPHERE	It creates a white 3D sphere of 1 unit scale.	<ul style="list-style-type: none"> <li>- Tick Interval</li> <li>- Repeats</li> </ul>
	CUBE	It creates a white 3D cube of 1 unit scale.	<ul style="list-style-type: none"> <li>- Tick Interval</li> <li>- Repeats</li> </ul>
	PERSON	It creates a 3D avatar that walks in space. There are three types of avatars selectable with the Model Index parameter.	<ul style="list-style-type: none"> <li>- Tick Interval</li> <li>- Repeats</li> <li>- Model Index</li> </ul>
	ANIMAL	It creates a 3D cubical animal that walks in space. There are 22 types of animals selectable with the Model Index parameter.	<ul style="list-style-type: none"> <li>- Tick Interval</li> <li>- Repeats</li> <li>- Model Index</li> </ul>
	MONSTER	It creates a 3D monster, an avatar that moves in space.	<ul style="list-style-type: none"> <li>- Tick Interval</li> <li>- Repeats</li> </ul>



ROCKET

It creates a 3D rocket.

- Tick Interval
- Repeats

Table 1. Emitter cube behaviors

### Transform cube



Figure 42. Transform cube

The Transform cube [Figure 42] is used to perform object transformations. It performs transformations in space and it also performs transformations to the object's properties, such as changing the color of the geometry.

The 6 behaviors attached to the Transform cube are the following [Table 2]:

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	FORWARD	It makes the objects move forward in the direction of the arrows.	- Speed






	ROTATE	It makes the objects rotate continuously around their own local axis.	- Speed
	SCALE	It scales the objects up or down.	- Scale
	UP	It makes the objects go up in space.	- Speed
	DOWN	It makes the objects go down in space.	- Speed
	COLOR	It changes the color of the objects.	- Color

Table 2. Transform cube behavior

**Split cube**



Figure 43. Split cube

The Split cube [Figure 43, Table 3] creates instances of the virtual objects and sends them in different directions. The Split cube relates to concepts such as Multithreading. The different duplicates act like threads on the program and each one follows its own different path simultaneously.

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	TWO SPLIT	It makes the objects split into two instances and follow the paths pointed by the arrows.	
	THREE SPLIT	It makes the objects split into three instances and follow the paths pointed by the arrows.	
	FOUR SPLIT	It makes the objects split into four instances and follow the paths pointed by the arrows.	
	FIVE SPLIT	It makes the objects split into five instances and follow the paths pointed by the arrows.	
	SIX SPLIT	It makes the objects split into six instances and follow the paths pointed by the arrows.	
	SPECIAL SPLIT	It makes the objects split into the amount specified by the parameter and go up.	- Number of Splits

Table 3. Split cube behavior





## Logic cube



Figure 44. Logic cube

The Logic cube [Figure 44, Table 4] is more related to mathematics and Boolean logic. It presents the concepts of logic gates that the user can apply to the virtual objects. This logic gates let some objects go through and apply a transformation to some other objects depending on an active flag that changes when the objects go through the cube. The frequency of the flag changing is controlled by a parameter called Divisor. Once the Divisor is set, an internal parameter Dividend is set to the Divisor value. The Dividend will decrease every time a virtual object goes through the cube. An internal division will be performed between the Dividend and the Divisor. The remainder will be used as a countdown before the next flag swap. As an example, if we set the Divisor to 7, 6 virtual objects will go through the cube, and the specific transformation will be applied to the seventh virtual object. Then the cycle will start over.

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	LOGIC GATE SIDE	It sends the objects forward and 90 degrees to the right in the frequency specified by Divisor.	- Divisor
	LAGIC GATE UP	It sends the objects forward and up in the frequency specified by Divisor.	- Divisor





	LOGIC GATE DOWN	It sends the objects forward and down in the frequency specified by Divisor.	- Divisor
	LOGIC GATE BACK	It sends the objects forward and back in the frequency specified by Divisor.	- Divisor
	LOGIC GATE DIE	It sends the objects forward and kills them in the frequency specified by Divisor.	- Divisor
	INTERPOLATE	It generates an object in the middle of two consecutive objects.	






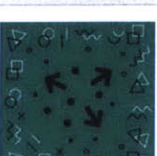
Table 4. Logic cube behavior

## Physics Cube



Figure 45. Physics cube

The Physics cube [Figure 45] contains behaviors related to real world physics concepts such as Gravity or Elasticity. Using the Physics cube, the child can experience different physics concepts applied to the virtual objects in space [Table 5].

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	GRAVITY	It changes the global gravity value on the scene.	- Gravity
	MASS	It changes the mass property of the object.	- Mass
	ELASTICITY	It changes the elasticity value of the object making it bounce more or less.	- Elasticity
	FORCE	It applies a force to the object.	- Force value
	ATTRACT	It applies an attraction force to the objects that are close to the cube.	- Force value
	REPEL	It applies a repulsion force to the objects that are close to the cube.	- Force value





*Table 5. Physics cube behaviors*

## Effects cube



Figure 46. Effects cube

The Effects cube [Figure 46] adds visual effects, music effects and animation effects to the objects and to itself. It is a cube meant for media effects for a playful experience [Table 6].

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	PARTICLES	It releases a burst of particles when the object collides with the cube.	
	SOUND	It plays a sound when the object collides with the cube.	- Sound Index
	ANIMATION	It changes the animation of the object.	- Animation Index
	PROPS	It activates the props that the object has.	



	TRAIL	It adds a trail to the back of the object.
	SHINE	It makes the object shine.


Table 6. Effects cube behaviors

### Gaming cube



Figure 47. Gaming cube

In order to give the children goals to achieve, we created the Gaming cube [Figure 47]. The Gaming cube contains different elements that can be used to generate a game scenario that the user can play with [].

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	CHECKPOINT	It lets the child specify a set of conditions that the objects have to satisfy.	Menu with conditions: <ul style="list-style-type: none"> <li>- Color</li> <li>- Speed</li> <li>- Scale</li> <li>- Type</li> </ul>






	COUNTER	Displays a counter of the objects that have collided with it.
	POINT UP	It gives a point to the object.
	POINT DOWN	It subtracts a point from the object.
	PORTAL IN	The objects that go through a Portal in will disappear and appear on the next Portal Out.
	PORTAL OUT	The objects that go through a Portal in will disappear and appear on the next Portal Out.

Table 7. Gaming cube behaviors

### Hyper cube

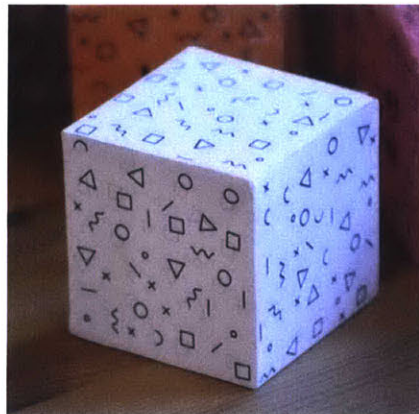



Figure 48. Hyper cube

The Hyper Cube [Figure 48] is a core cube that serves as a box to keep other behaviors and scenes. With the Hyper Cube, the user can create scenes and save them to load them later on a new environment. It allows the saving of processes in space. It is directly related to the term encapsulation.

With the HyperCube, the user can save a previously generated scene. The user can later reload the scene and all the cubes will appear as digital instances in space. The virtual objects will react to the digital instances of the cubes as if they were physical.

GLYPH	BEHAVIOR	DESCRIPTION	PARAMETERS
	HYPERCUBE	Saving and loading virtual scenes previously generated.	<ul style="list-style-type: none"> <li>- Load Scene</li> <li>- Save Scene</li> </ul>

*Table 8. Hyper cube behaviors*

\*All sides of the Hyper Cube have a similar pattern and have the same behavior attached to them.

## 4.2.2 The cycle as a programming loop or main function.

The HyperCubes application acts as a programming loop, repeatedly invoking a starting point as a timed operation. From the Emitter Cube, it creates virtual objects that get generated in a loop and move constantly forward when a Forward behavior is used. This emitting process is similar to a sequence in a program that runs endlessly. Various types of programs work with an infinite main loop such as the Processing (Fry & Reas, 2003) or VVVV (VVVV group, 1998) platforms. In our case, the loop cycle is initiated with the Play button in the interface and it goes on forever or until the user stops it with the same button. The flow of objects act as a sequence of a program running through different lines of code. In HyperCubes, these lines of code are no longer text but physical tokens that the child can manipulate and move in space.

In this example [Figure 49] the user makes use of the Emitter cube and the Transform cube to generate a sphere and make it move forward in space.

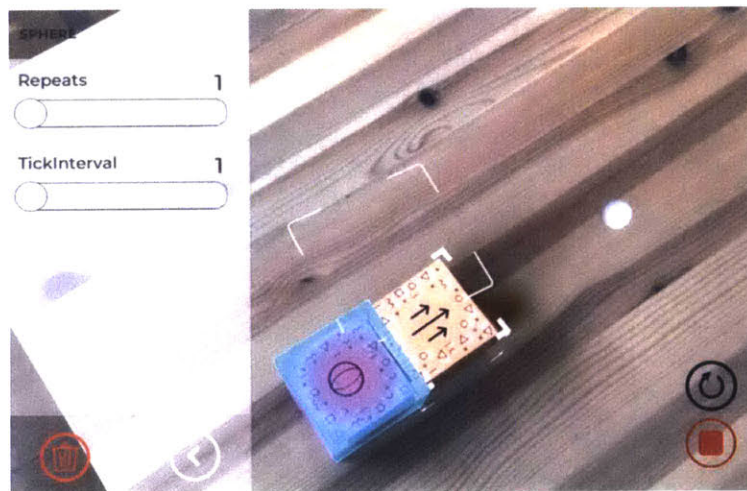


Figure 49. HyperCubes: Emitting one sphere

Every time an object collides with a cube, such as the Transform cube, the cube lights up to signal its activation. This activation happens with all the cubes to assist the user in understanding the sequential nature of their program. The sphere moves in space continuously, following a straight path and the user can move in space too, following the sphere.



### 4.2.3 Cubes and Parameters as Functions and Variables

The cubes act as encapsulated functions that the child can use along the path of the virtual objects. When a cube is selected, a GUI shows a menu to modify the parameters of that behavior instance. These parameters are an introduction to the concept of variables that the user can play with. The user can observe how they affect their behaviors and helps them achieve different outputs.

Using the parameters in the Emitter cube, the user specifies how often each sphere will appear and how many times this event will happen. In the Transform cube, the user tweaks the speed at which the spheres will move.

In this following example [Figure 50], the child makes use of the Emitter cube and the Transform cube to emit people. It generates a trail of little people walking in space. The user tweaks the variables on the GUI menu to make the emission repeat forever in a tick interval of 1 second. The user can move the camera away from the cube and still see how the little people move in space.

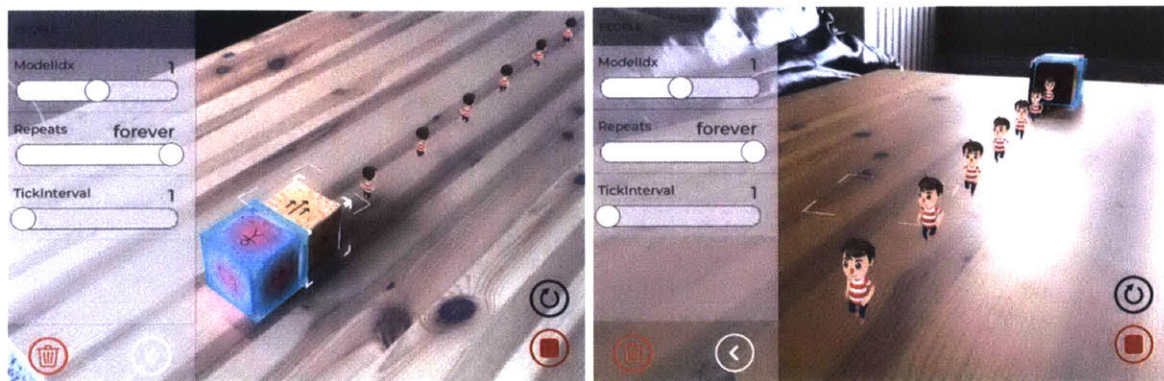


Figure 50. HyperCubes: Example with an Emitter cube and a Transform cube emitting people

In this other example [Figure 51, Figure 52], the child emits little deer avatars. It makes use of a Split cube to duplicate and spread the paths of the little animals in space. The child positions a Transform cube in one of the paths in order to change the color of the deer. The child uses another Transform cube to modify the scale of the deer in one of the split paths.

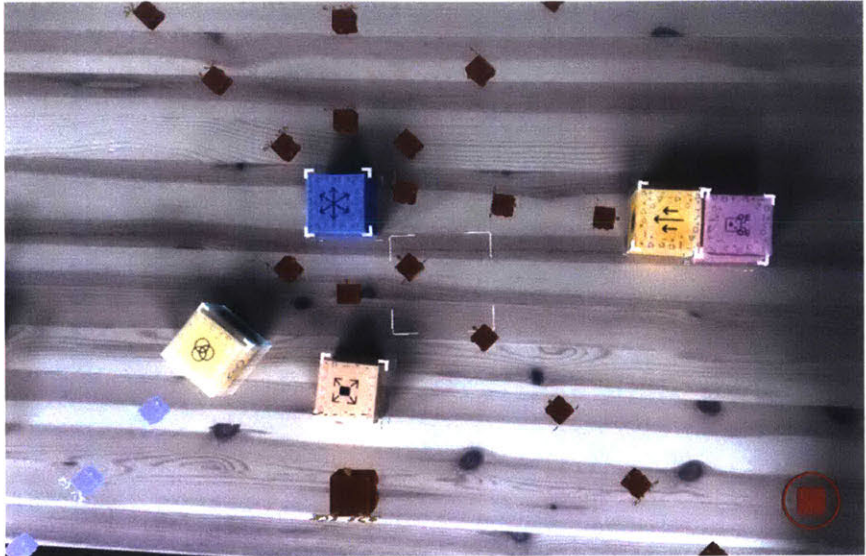


Figure 51. HyperCubes: Aerial perspective. Split and Transform cubes

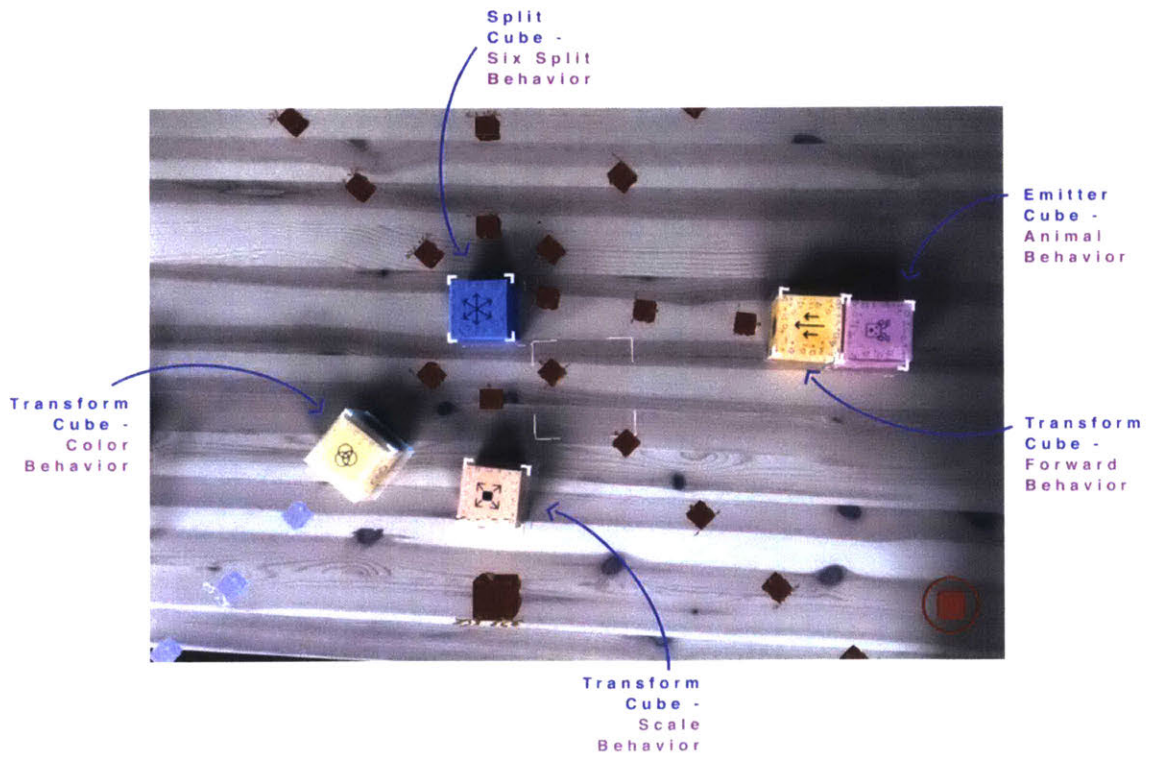
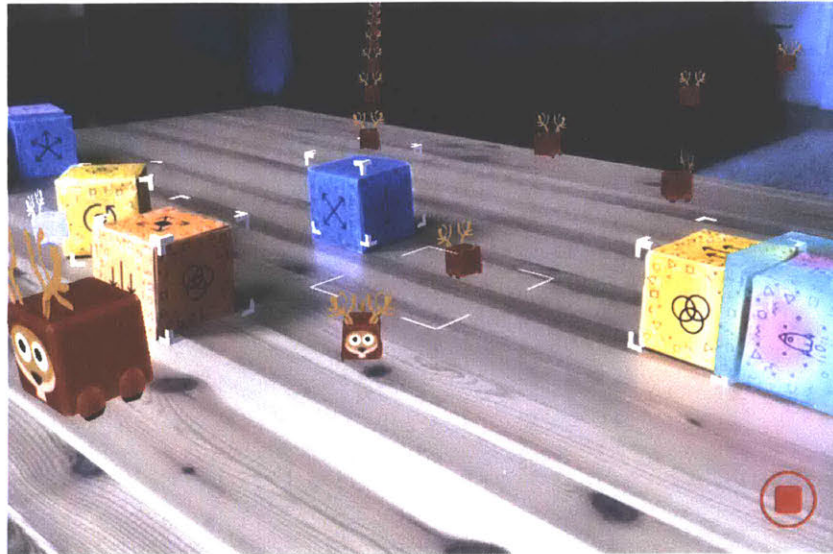


Figure 52. HyperCubes: Aerial perspective. Scene explained

The same scene seen from another perspective [Figure 53] can be observed here:



*Figure 53. HyperCube: Side perspective. Split and Transform cubes*

Another example can be observed in the next figures [Figure 54, Figure 55]; the child plays with the Gaming cube in order to create a game. The child sets a couple of Checkpoint behaviors and creates different conditions. The conditions can be added through the GUI and all the objects that go through that cube will have to satisfy the set conditions. In this case, the Checkpoint being set up expects 10 blue spheres with a scale of two units. The user can add or remove conditions as wanted.

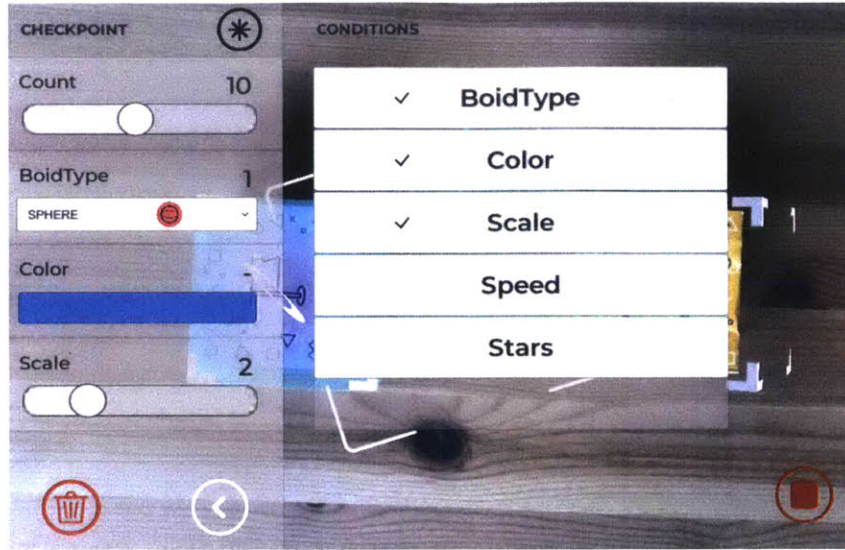


Figure 54. HyperCubes: Conditions Menu on Checkpoint behavior

Later on, the child can hand in the iPad to a friend and ask this other friend to solve the Checkpoints for him adding more cubes to accomplish the different conditions.

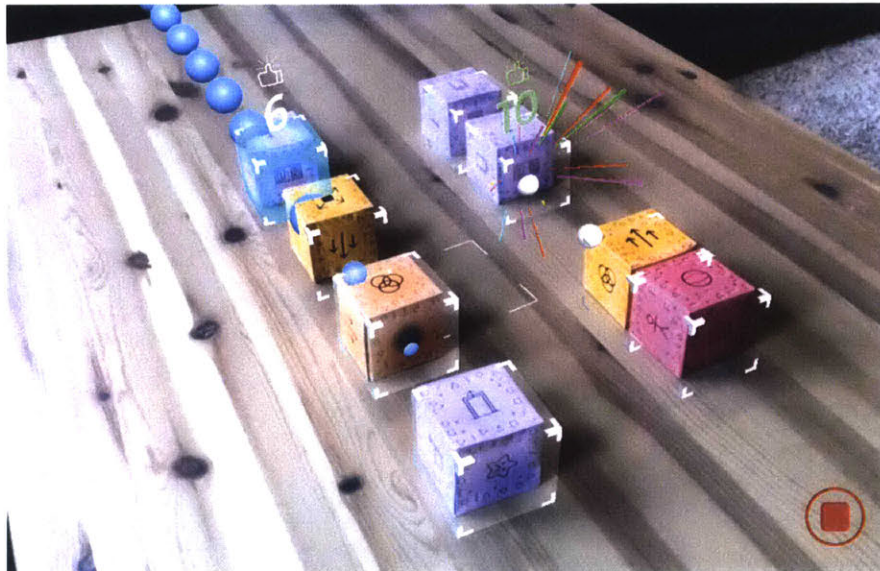


Figure 55. HyperCubes: Gaming scene being resolved

The person trying to solve the puzzle makes use of Portals, Color and Scale behaviors in order to solve the two Checkpoints. In the image we can observe how the first Checkpoint has just been solved (the celebration particles appear when a Checkpoint is resolved) and the second Checkpoint has gotten 6 correct virtual objects so far.

#### 4.2.4 Cubes as Programming Functions and Events

The cube's active behavior is defined by the side that is oriented upwards. When each digital object collides with a physical cube, an event is triggered. The object exits the cube having gone through a specific transformation. Each one of these behaviors executes as a function applying its encoded procedure to the object. Different programming functions are physically embodied in each one of the cubes. The variety of behaviors and their different implementations relate to the concept of polymorphism. The main idea is not requiring the child to understand what a programming function is but giving her an initial platform to tinker with a physical representation of all of these concepts. The goal is to let her discover how certain processes will create different outputs depending on their sequential order.

Using Sound behavior, a child can create a song by configuring the cubes in the proper order. In this next example [Figure 56, Figure 57], the user utilizes a Split cube, a Logic cube and some Sound behaviors to create a song.

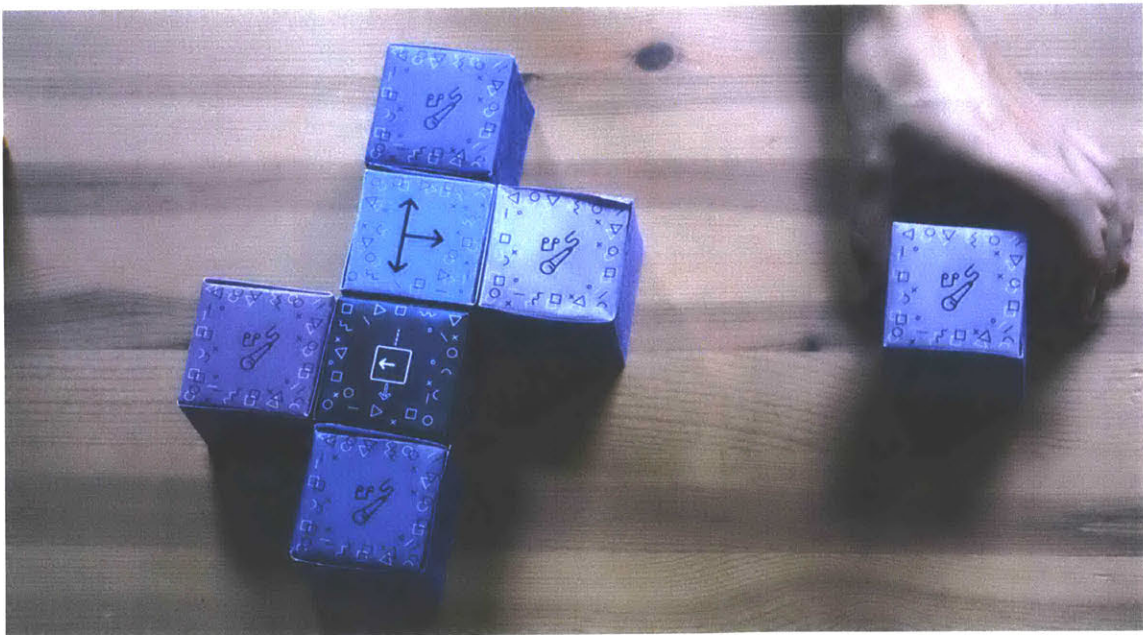
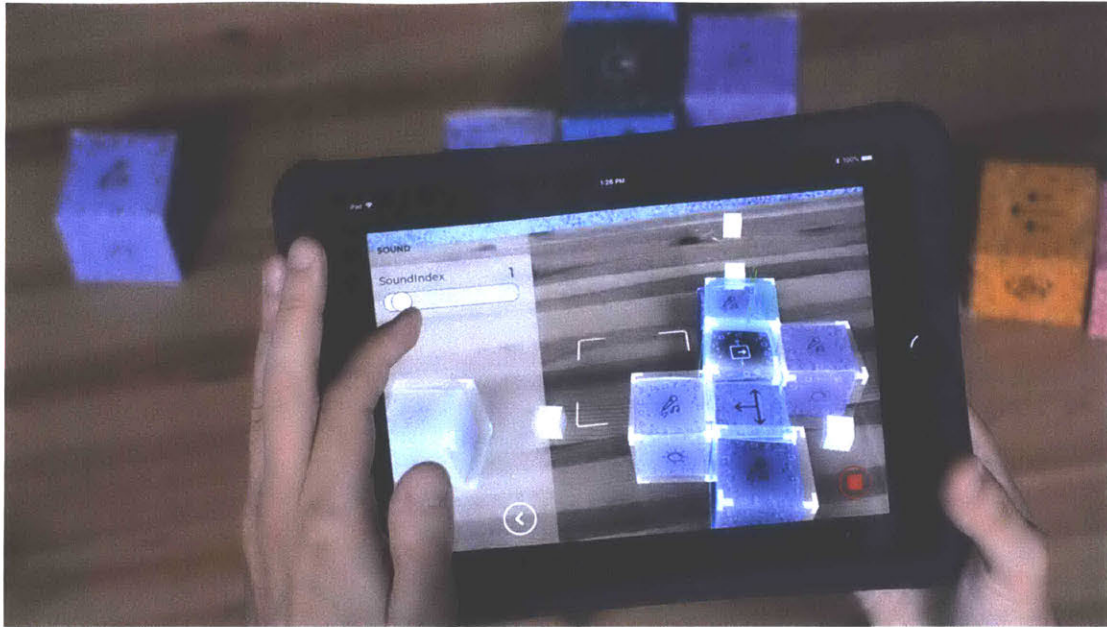


Figure 56. HyperCubes: Configuration of cubes in space



*Figure 57. HyperCubes: User creates song with Sound behaviors*

With the Logic cube, the user can make the flow of the objects asynchronous. This way, interesting configurations can start to give different outputs when using sound for example. In this specific example, the user is placing two Sound behaviors at the same distance in order to create a chord. The Logic cube allows for a specific amount of virtual objects to go through at a time. With this, the user can get different rhythms. Changing the “Sound Index” parameter on the Sound behavior, the user can get different notes and tones. Making use of the cubes in space, the user gets different sound outputs that can generate a song.

### 4.3 HyperCubes as an authoring tool for creativity



Figure 58. HyperCubes paper markers

#### 4.3.1 Playfulness and creativity for learning

HyperCubes not only gives a framework for the children to author and create but it also provides a playground that makes use of the basic materials and physical elements that children are used to play with. Embodiment and spatial cognition are essential in the real playground and they are too in HyperCubes. The “Creative Learning Spiral” from Mitch Resnick is leveraged to improve spatial abilities and to relate them to computational thinking in the physical environment and in a playful way. The Gaming cube lets the user design and create challenges following the “Puzzle it out” design pattern that Case highlights in his review (Case, 2018). With Checkpoint we give the child the opportunity to create challenges and to engage with each other to try and solve them using logic and reasoning.

#### 4.3.2 Low floors and high ceilings

HyperCubes is targeted particularly to 8-12 year olds. Nonetheless, the application is very scalable and can be used in different contexts. A young child may make virtual objects appear to understand what the Emitter and Transform cubes do. An older child can create a song by using the Sound behavior and the Logic cube to make virtual objects move asynchronously or could even create a binary calculator by using the logic gates of the Logic cube. This broad spectrum of options and difficulty for different age levels is what Seymour Papert called “Low floors and High ceilings”. The application covers a wide age range. It is accessible to all learners. Each of the age groups will engage and play with the application in different ways more suitable to their knowledge and educational standards.



### **4.3.3 Collaboration in space**

One of the valuable aspects of HyperCubes is the fact that children play together with the application. They do not interact alone with a mobile device but they work together to achieve goals, moving the different cubes on the surfaces or interacting with the smart devices together. They have conversations and discussions on how to accomplish certain tasks. HyperCubes was designed in order to allow for collocated interactions in a common physical environment. The idea of using a smart device individually seems alienating and can come through as a flaw of the system; however, after carrying out some user studies, teachers and researchers were surprised at the level of collaboration that happened during the workshops.

## 4.4 System overview and implementation

### 4.4.1 Design and UX

#### *The cube patterns*

In order to design the cubes, a set of iterations were performed where different designs were generated and tested. The main requirements for the Augmented Reality recognition algorithms to work make the design process more challenging when it comes to getting a nice and coherent output. The AR library stores the set of image targets for each cube and generates a file that is imported into the software platform to be recognized and the object tracked. The image patterns have to have a set of characteristics in order to make the recognition easier. In order to track the position, rotation and scale of an object the patterns require distinct attributes that can be tracked at distances and skewed angles. They have to be contrasted and with angular shapes [Figure 59].

The final design is cohesive through all of the cubes and it is based in a pale color palette [Figure 60]. The design of the patterns for each cube is based on a master pattern that is randomized for each side of the cube and used as a frame surrounding the main glyph [Figure 61].

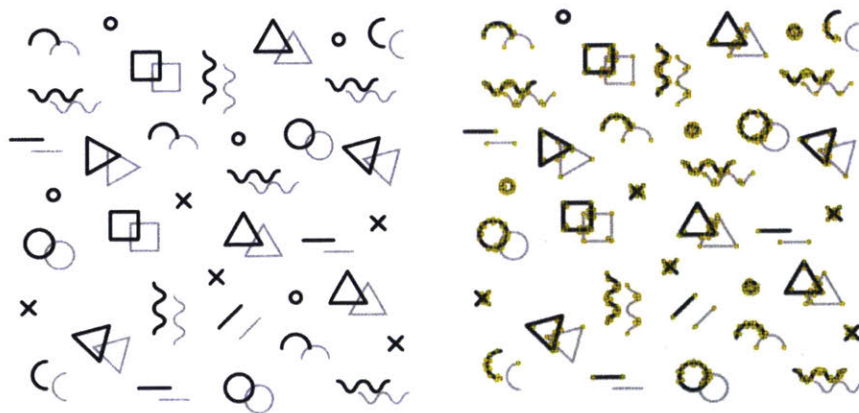


Figure 59. From left to right, pattern used for the design of the cubes and features (yellow marks) recognized by the AR algorithm

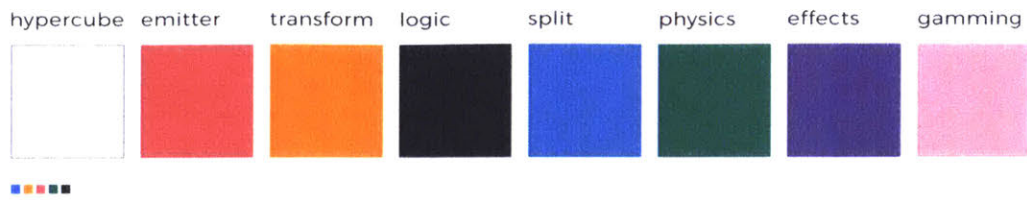


Figure 60. Color palette selected for the set of cubes

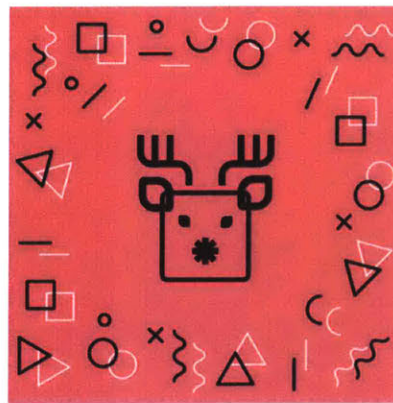


Figure 61. Cube side example with pattern surrounding glyph (Animal behavior)

### ***The graphical user interface***

The parameters and different options of the application are tweaked and controlled through a main GUI. This GUI has to be light and not obstruct the AR content. This way, we don't block elements of the physical environment and we keep the user engaged beyond the screen. The solution used for HyperCubes was to design a menu on the side with a low-opacity background [Figure 62, Figure 63]. This design creates an ideal user experience by keeping the buttons close to the edges of the device. By keeping controls close to the device edge and near their fingers natural resting place, they are easier to adjust while holding on to the device and moving around at the same time.

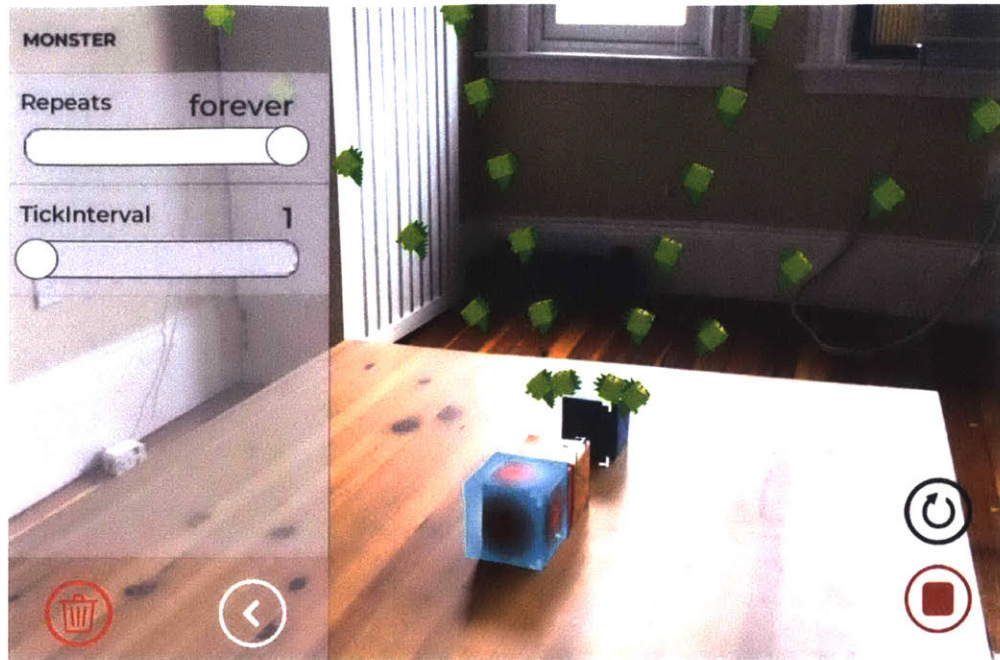


Figure 62. HyperCubes GUI: Editing Emitter Parameters.

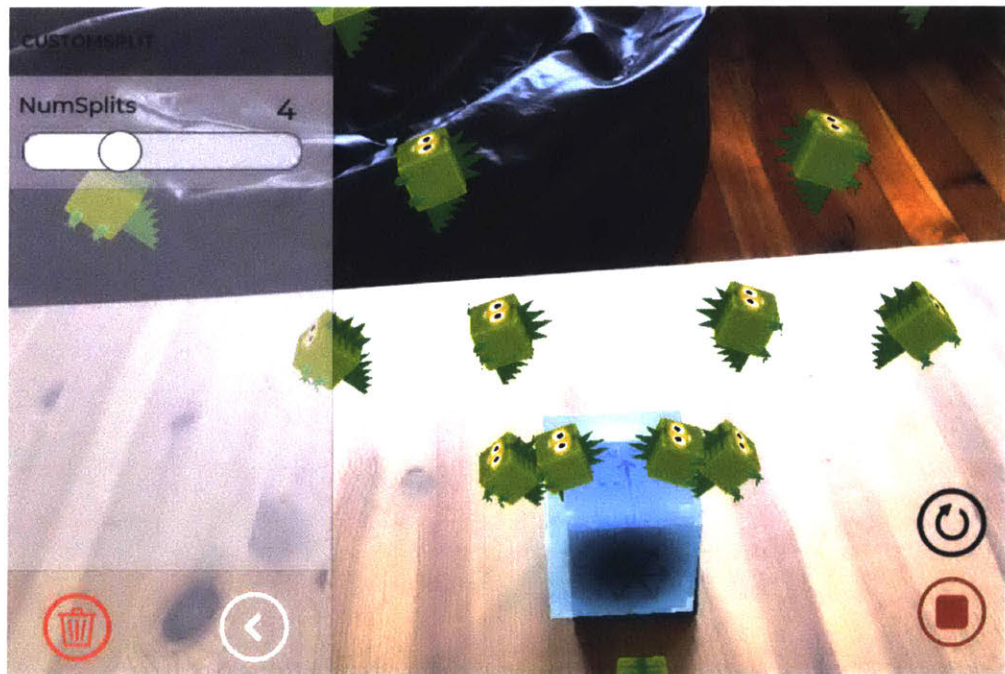


Figure 63. HyperCubes GUI: Editing CustomSplit Parameters.

### ***The user experience***

The user interaction within the Augmented Reality application has to allow for a natural manipulation of virtual and physical elements while holding the smart device. When designing AR interfaces with tangible elements for mobile devices it is important to design for the user to be able to hold the device with one hand while performing actions with the other hand. Holding the device severely limits the functionality of one of their hands and thus it reduces the interaction possibilities. Using a mobile device lets the child release the device at any moment and play with the physical cubes on top of a surface in order to configure them. It is important to design an experience where later on they can pick up the device again and interact with the AR application in the manner they had previously. The user should feel unconcerned with moving between editing the real world and her digital world.

## 4.4.2 Main Software Architecture

The software platform used to develop the application is Unity3D. Unity3D is a popular game engine that offers a lot of flexibility and valuable tools when developing applications that make use of 3D virtual content. The Augmented Reality content is tracked and displayed using the Vuforia library for Unity3D.

HyperCubes is programmed using Microsoft's C#; it is a strongly-typed programming language that supports object-oriented patterns. Using Classes supported the development of an approachable inheritance model. Language features such as Attributes (*also referred to as decorators*), Generics and Reflection were a good match for the project. Using these features we were able to design the different variables for each Behavior in a declarative fashion and enable the parameters interface to be generated uniquely for each one based on the Reflection's capturing of each variable type and its Attributes. With Reflection classes the public variables are exposed and if, for example, the variable is a number with an Attribute stating its minimum and maximum values, a slider will be rendered to the Canvas. This way, the GUI has access to all the variables from the Behaviors, can draw them and modify them. This enabled faster prototyping and exploration of valuable parameters as well as consistent progressive enhancement of design.

To aid in the prototyping and development of HyperCubes a desktop emulator was created [Figure 64]. The emulator runs on the Unity3D platform targeting Mac and Windows and doesn't require any Augmented Reality software. It is an application to configure and test the cubes on a regular computer used for development and leverages Unity3D's built-in scene editor for live-editing objects. The cubes are created only in the virtual environment mimicking the laws of the physical setting when getting detected by the AR application. The affordances of working on a standard computer with provided development tools allowed for a faster development of the final AR application.

With the emulator the user can perform the same actions as with the physical cubes and the AR application. It has the same graphical user interface to tweak parameters and it has a menu with all the cube types in order to add them to the scene. The emulator allows the user to load and save scenes previously created.

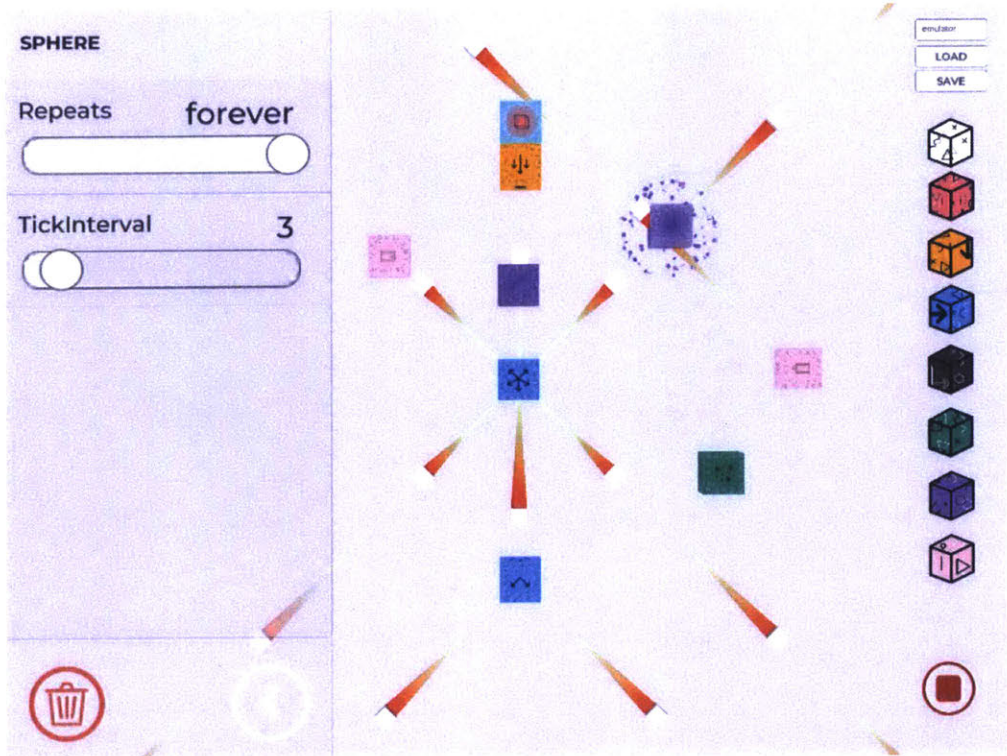


Figure 64. HyperCubes Emulator

### 4.4.3 Augmented Reality technologies

With the Vuforia library, we can detect the markers in the physical environment. It performs tracking of different objects or images. In this case, the tracking of the cubes was done by recognizing the sides with the pattern designs already shown. With the Vuforia Target Manager, the patterns can be uploaded for each side of the cube creating a Multi-Target. The Vuforia online application stores this patterns and gives an estimate rating of the tracking quality of each image target. The image targets have to be contrasted images in order to get a good tracking. For HyperCubes, only images that gave excellent tracking quality were used in order to avoid having problems with certain light conditions [Figure 65].

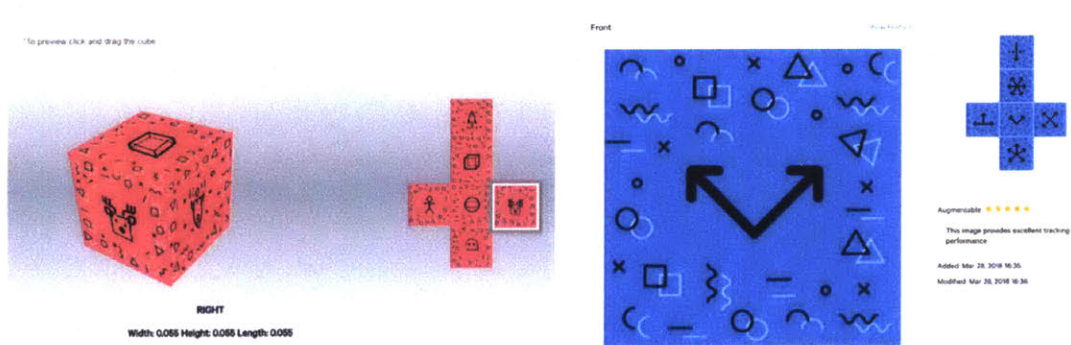


Figure 65. From left to right, Vuforia online application to load Multi-Targets and an Image Target tracking rating

Surface (or Planar) Tracking detects and maps flat surfaces such as a table or floor. To perform Surface Tracking, the ARKit and ARCore libraries are used. ARKit is a library used for Apple's operating systems while ARCore is used for Android. These two libraries are incorporated into the Vuforia library and can be used in Unity. This way, the multi-targets can be tracked and anchored in space. The content becomes persistent. If the user moves the camera so that the object is no longer visible, the digital content won't disappear. The object will remain anchored to the space that is still being tracked [Figure 66Figure 67].



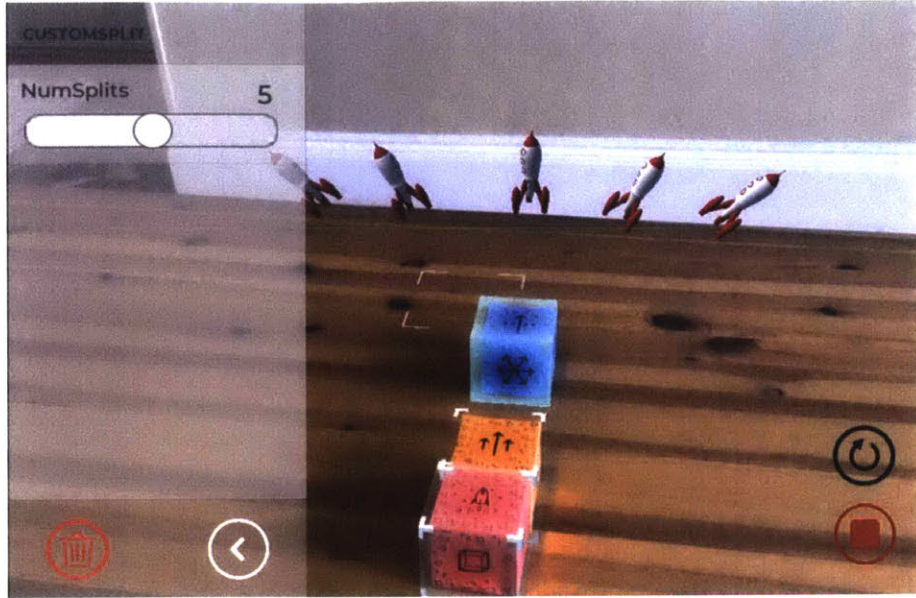


Figure 66. HyperCubes: Generating Rockets and splitting them up in the air.

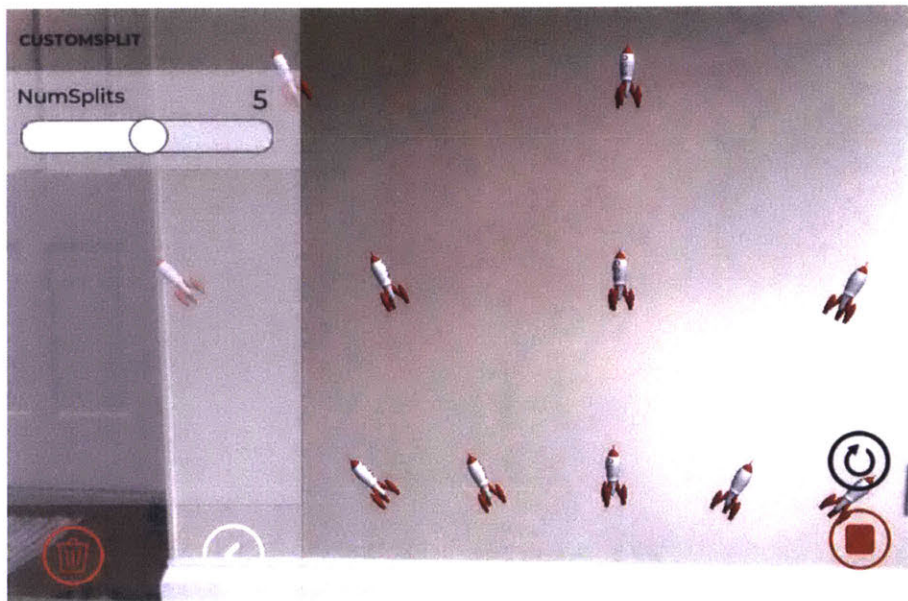


Figure 67. HyperCubes: When pointing upwards, the virtual content persists even if the cube are not in view.

## **4.5 Evaluation**

In order to test the project, a qualitative evaluation was conducted. A first pilot study was organized with children between 9 and 12 years old. Later on, a qualitative user study was conducted with children of ages 8 to 13. The pilot study served as a first test to get a feel for the usability of the application and the engagement levels of the children. This first pilot study helped in getting a lot of constructive feedback and positive observations from children and tutors. After the pilot study, improvements to the application were developed. The experience was very useful when preparing the software and the test for the qualitative user studies. The qualitative user studies gave a lot of positive feedback and good responses that validated the engagement and interest towards the application.

### **4.5.1 Pilot study**

A pilot study was performed in Leicester, United Kingdom, in the context of the Leicester Art-AI Festival. A set of 3 workshops were performed on the same day with three groups of children from the Sandfield Close School [Figure 68]. Each workshop had a duration of 1 hour and 30 minutes. 15 children participated in each one of the workshops having a total of 45 children engaging with the application. In the pilot study both Paper Cubes and HyperCubes applications were tested. All children and parents had signed media release forms. The procedure of the workshop was the following:

- 10-15 minutes of explanation of the project.
- 30 minutes of mounting paper cubes.
- 30-40 minutes playing with the AR application.
- 10-15 minutes of feedback and comments.

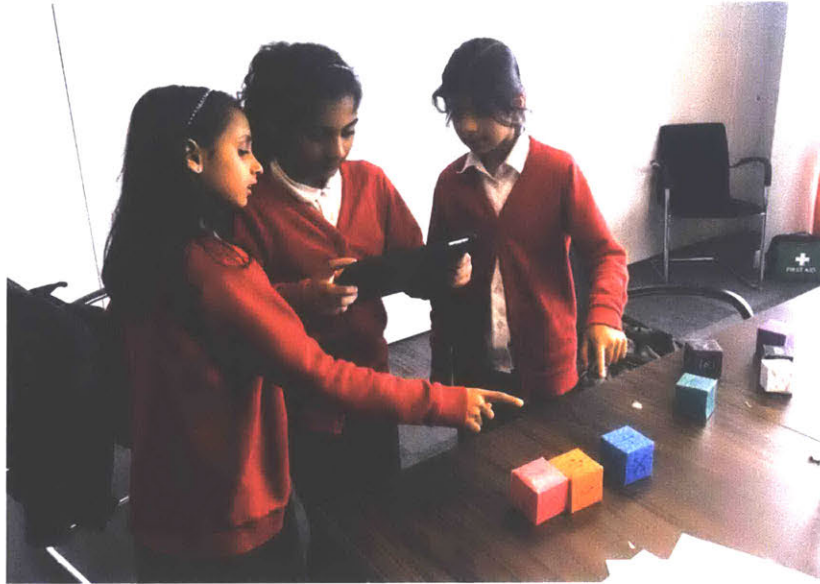


Figure 68. HyperCubes at the Leicester Art-AI Festival

At the end of the workshop, a 10-minute session served to let the students and teachers give feedback on the applications. They all wrote down notes on post its and read them out loud for everyone to listen [Figure 69].

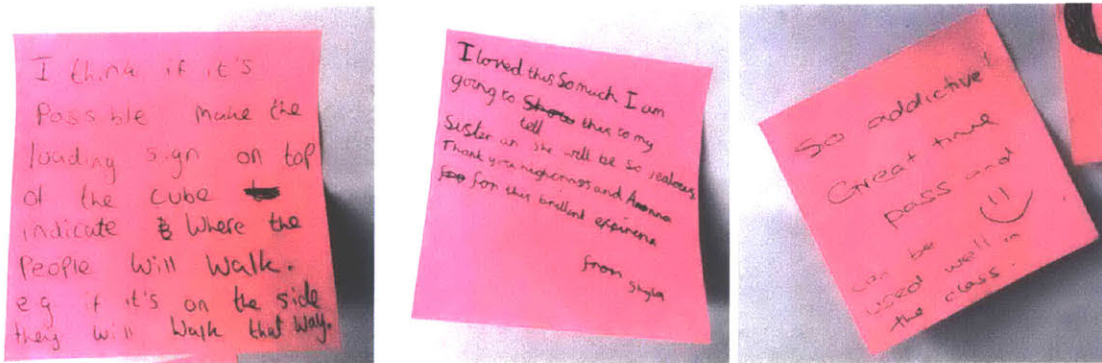


Figure 69. Samples of post-it feedback from teachers and students

The tutors of the students were present during the workshop helping the children and engaging also with the application. A survey was given to them at the end. This following

graph [Figure 70] shows the median results of the survey realized by the 4 tutors that accompanied the children:

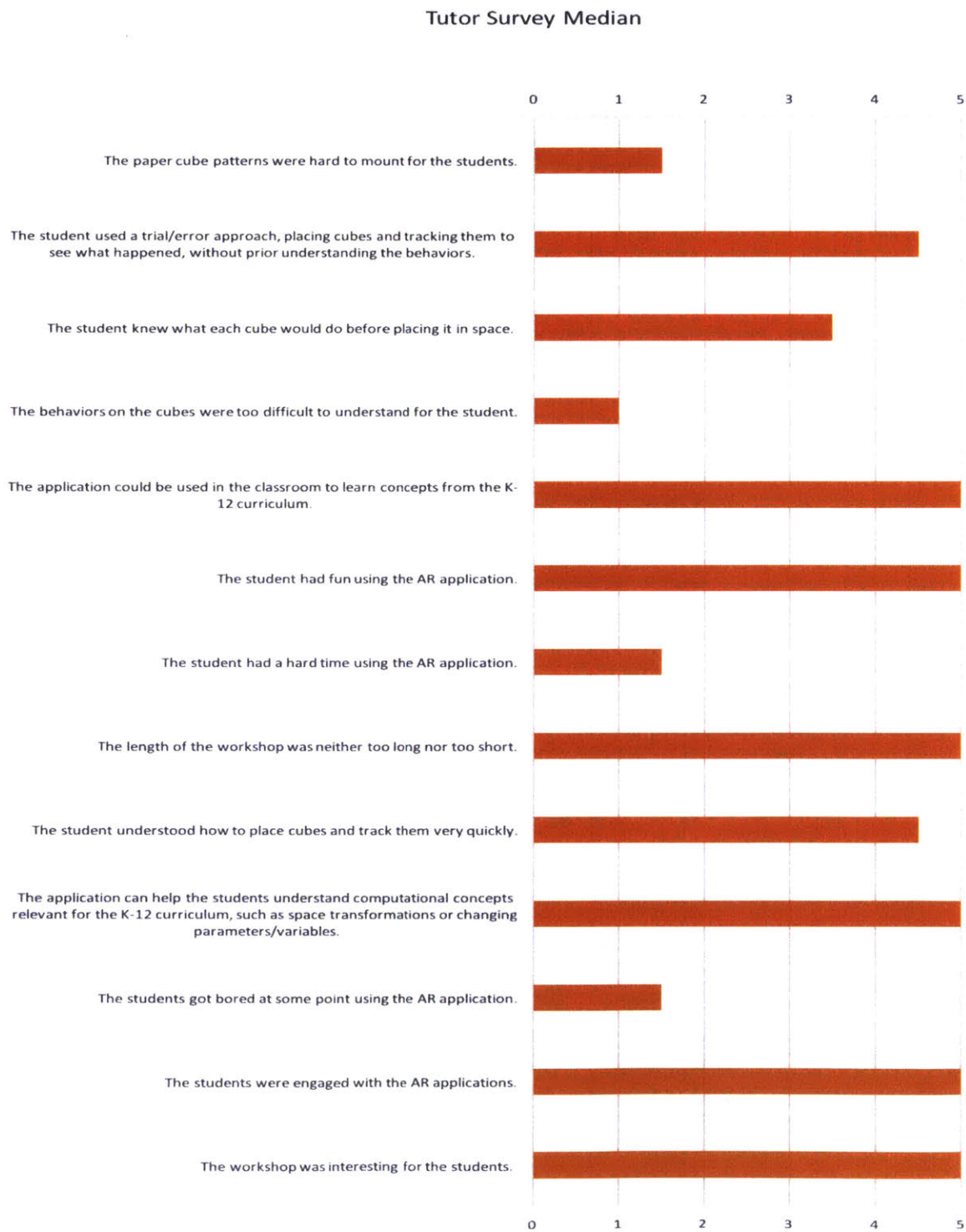


Figure 70. Tutor Survey Results

The surveys also had questions to answer. Some of the most interesting answers from the tutors were the following ones [Figure 71]:

- Which cube did the student like the most? Why?  
*'HyperCubes because it was easier for them to manipulate and arrange to make the stickman move'*  
*'Effects cube – lots of different variations'*
- Which cube was harder to understand by the student? Why?  
*'Physics cube – understanding of terminology for some of the children'*
- What do you think could be improved in the application to be valuable for the student?  
*'Links to mathematics and English, e.g. multiplication tables'*  
*'Maybe show it on a bigger screen for students to see'*  
*'Understand each cube (beforehand) and how far the phone needs to be from the cube'*  
*'Some cubes have no words on which means the pupils had to use trial and error'*
- What concepts from the application can you see as useful for the K-12 curriculum?  
*'All of it'*  
*'Programming, using logic, designing an algorithm'*  
*'The commands. Understanding why its moving the way it does'*
- What classes/courses could you see this application being used for?  
*'ICT, Literacy, Social communication'*  
*'ICT, Team building'*  
*'Maths, programming/computing, Literature – creative writing'*

Figure 71. Relevant responses from the tutor's survey

The students and teachers were later interview by local media. These are some of the excerpts from the interviews [Figure 72]:

WORKSHOP FEEDBACK:

Interviewer: *What's different about using computers this way?*

Dylan, 10 years old [student]: *'This one is much easier for children cause computers they normally have loads of buttons and apps. With this all you need to do is tap it and then you get all these animations which is quite cool'.*

Interviewer: *Is it more like a game than programming?*

Dylan, 10 years old [student]: *"It's sort of both really. You get sort of addicted to it, like my teacher did"*

Mrs Eammes [teacher]: *'The program that they are using builds upon the knowledge of what it is learnt in the first cube and you can put different cubes together and come up with different outcomes so children is communicating with each other: "have you tried this? Have you found a sheep? How did you do this?" so it's a lot more interactive than I first thought. I thought it might just be the children sitting there with an iPad or a phone and it would be a very insular thing. Whereas actually them talking and communicating with each other it's been really good to see.'*

*Figure 72. Exerpts from the media interviews to teachers and students at the Leicester Art-AI Festival*

The take-aways from the pilot study were the following:

The engagement levels were very high, all students played with the applications and several of them asked for the public release of the application in order to play at home. All of the students took the cubes they had mounted with them when the activity was done.

What most surprised teachers and researchers was the level of collaboration between the students when using the applications. They played in groups, discussing what to do next and trying to achieve certain goals together.

All teachers agreed that the application could be helpful for the classroom and several class subjects could benefit from using it. Some of them pointed out the need for more feedback in the behaviors on the cubes so that the interaction was more intuitive.

After the pilot study, several improvements based on the feedback were incorporated in the application. The elements on the GUI were made bigger in size to facilitate the interaction, especially with buttons and sliders. More visual feedback was added in order to understand when the different events happened. Better tracking feedback was added to the cubes.

#### 4.5.2 Qualitative user study

A second qualitative user study took place at Parts & Crafts, in Somerville, Massachusetts in the context of the Girls Invention Week summer camp [Figure 73]. HyperCubes was presented as a voluntary activity during two days at the summer amongst many other activities. The activity ran from 10am to 11.30am both days. Children could decide what activities they would engage with and were not forced into any of them. Other types of activities apart from HyperCubes were movie making, sword making, screen printing, playing with water or making a fort. The fact that children would come and play with the application over other activities would validate further the engagement with it and the interest in it. The children that wanted to play with the application were given the iPad, a document with the application explained and the cube patterns. Mounting cubes was a voluntary activity. They were provided with a set of cubes already mounted in case they wanted to play with the AR application straight away. The researcher observed and took notes of the interaction without intervening in the activity if the children didn't ask for help.



Figure 73. HyperCubes at the Parts&Crafts Girl's Invention week



The take-aways from the user study are the following:

### **Age target**

*'It looks like younger kids (9-12) want to play more time than older kids (12-14).'*

Younger children are more engaged with the application in general. Early teenagers get tired of playing with it faster. Although there are more young children in the group and only a couple of teenagers. The group count may not be significant enough to extract conclusions on age targeting.

### **Interaction with smart device and physical environment**

*'The iPad is heavy'*

*'Sometimes, when they reach for the cubes while holding the iPad, looking through the camera, they cannot synchronize the hand with the position of the cube. The iPad is too big and the camera has a big offset with reality.'*

*'More behaviors and parameters would let them create more.'*

During the activity it is clear the fact that the iPad is too heavy for some of the children. Sometimes it is hard for them to hold it with just one hand. There is also an interaction offset when they try to grab a cube with one hand looking through the iPad.

### **Interest and engagement**

*'They are very impressed and use words like 'cool' or 'awesome' when they see the app working for the first time.'*

*'Students that played with HyperCubes the day before come back as soon as I put the cubes out.'*

*'Several of them ask for the app to take home.'*

The engagement levels are very high. The first day of the activity, around 10 children stayed in the HyperCubes area. They take turns. As one plays with the application the others mount

the paper cubes. The second day of the activity, a lot of them come back to play with it again. Some of them even spend more time than the day before.

### **Basic Programming Concepts**

*'All of them understand the LOOP concept. They mention Scratch and coding.'*

*'They understand very quickly how to use parameters in order to change the variables on each cube.'*

*'They understand computing behaviors and LOGIC behaviors.'*

Throughout the two days several moments and situations indicate that the children understand the basic programming concepts behind the different behaviors and parameters. They quickly understand how modifying parameters will get them different outputs and they are able to reason and change the parameters that will get them what they need.

### **Understanding sequential events**

*'One of the student places a COLOR behavior after a SPLIT. She stops to think why not all of them have color. Then she understands that she has to place it in front of the SPLIT in order to get all of them with the same color.'*

*'One of the students wants to make the boids go up and then down. She puts the BALLOON first, and all of the boids go up. She puts the DOWN behavior after the UP behavior but the boids are going up, so they never reach the DOWN behavior. She then realizes she can use a MODULUS behavior to make one boid go up then another boid go forward and down.'*

Several situations with different kids show how they make an exercise to understand the sequencing nature of the application in space. They place the cubes in different order and stop to think why the output is the way it is. They modify the configuration in order to get the output they want after reasoning and understanding what is happening.

### **Use of Space**

*'They make use of space to make different things happen. They place PORTALS in different spots on the room.'*

*'She also stacks the cube up and places them on top of a toy tower in order to split the boids up in the air.'*

*'They try to track cubes in space, holding them with their hands. And they stack them to get to the boids that go up with the BALLOON.'*

They make use of space constantly, making the boids go up and playing with Portals to spread the content all over the room. They watch the boids collide with other children in the room. They also stack them using other objects that are laying around on the table.

### **DIY and tangibility**

*'A lot of them want to make cubes: cut the patterns and mount them. One girl makes a whole set of them.'*

*'They play with the physical cubes once they have them built. One girl tries to juggle with them.'*

Mounting cubes is a voluntary activity. They do want to mount cubes and some of them spend more time mounting cubes than playing with the application. They want to take the cubes home and ask later for the availability of the application to play at home.

### **Collaboration**

*'They debate in groups and try to argue what will happen and what cubes they need to use in order to achieve a certain goal.'*

As seen in the pilot study, the students play together with the application and discuss what to do next in order to achieve certain goals. Some of them play alone when there are not that many students. Some of them play together and try to collaborate between them to achieve what they have agreed at the beginning.

### **Puzzle it out**

*'Several of them like to create CHECKPOINTS and solve them themselves.'*

*'They try to think about the output in advance. They use reasoning to project and think about what will happen.'*

They like the Checkpoint behavior considerably. They like to set up conditions and then solve the Checkpoint themselves by looking for the appropriate cubes.

### **Place your bets**

*'They (...) try to argue what will happen and what cubes they need to use in order to achieve a certain goal.'*

They project to the future and try to guess what will happen when placing a certain cube in a certain position.

### **Simulations**

*'The different models and effects make them very engaged.'*

*'Especially the ANIMALS. They go through all of the animal models and try to get specific ones, giving them color and splitting them.'*

They are very interested specifically in the different avatars and the visual and sound effects generated by the application. They particularly like the Animal avatars that get them very engaged. They also like creating music and they sometimes dance to their own creations.

Finally, a last qualitative user study was performed at the MIT Museum in the context of the IdeaHub program. A workshop was performed from 10am to 1pm the 25<sup>th</sup> of July of 2018. Children and parents could walk in and engage with the application voluntarily. They all signed media release forms and consent forms. In this last study, similar interactions were observed from the Parts & Crafts study. They could choose whether to play with the application straight away or mount the cubes first. Most of the users decided to mount cubes first. We could witness the engagement of the parents playing with the children and we captured interactions from younger and older users.

The qualitative user studies gave us a sense of the level of engagement and interest that the application generated. The user studies are not conclusive on learning rates by students. In order to get conclusive results on learning rates, a deeper quantitative user study should be performed with more time and resources. In this case, we proceeded with a qualitative exploration that gave us good results to show the potential of the application. At this point the application is ready and stable enough to perform a quantitative evaluation in order to get further results.



Figure 74. Children mounting cubes and cubes mounted

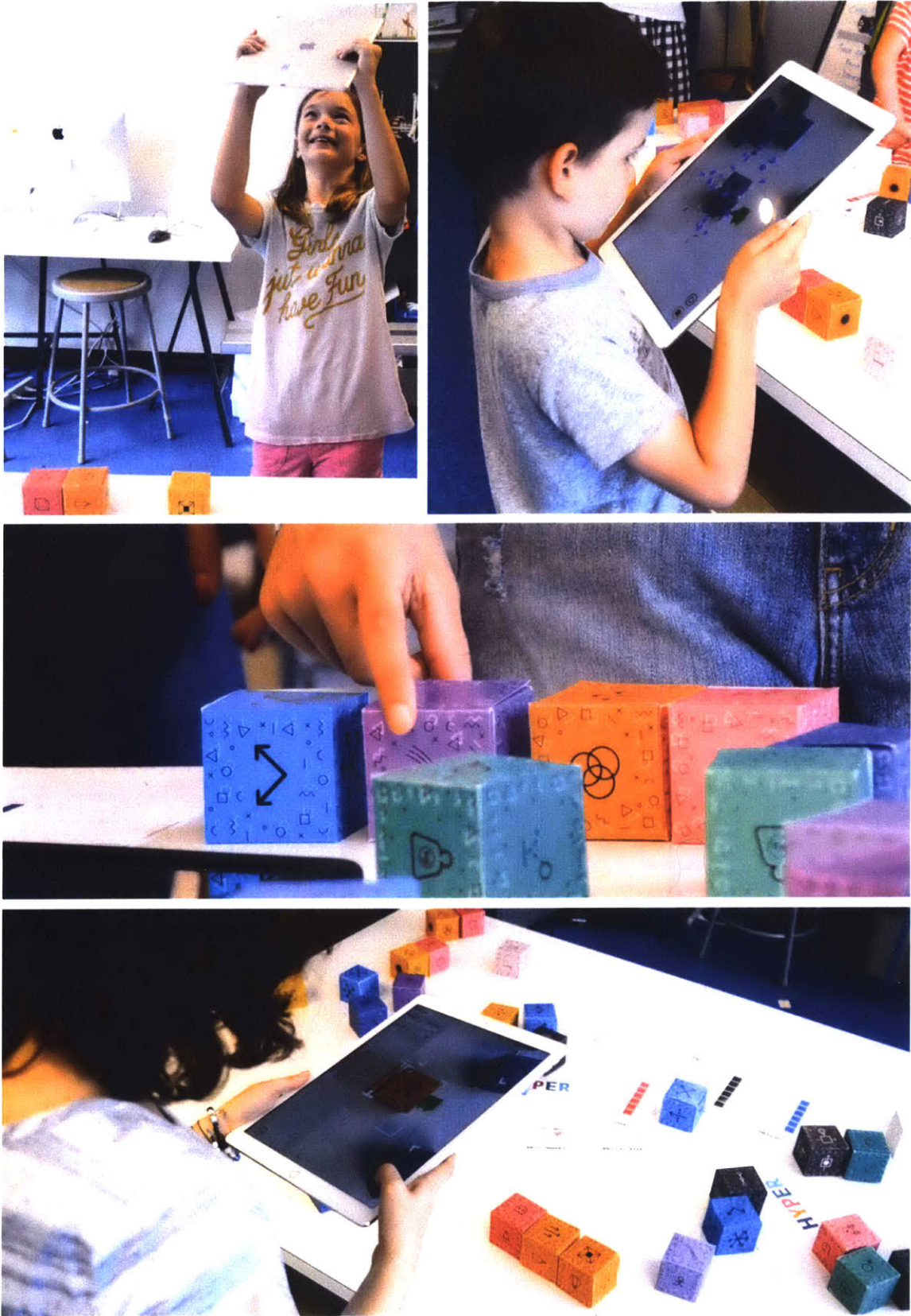


Figure 75. Children playing with HyperCubes at MIT Museum

## 5 CONCLUSIONS

The first research goal of this thesis was to propose a new interaction framework for learning computational thinking making use of space and the physical environment of the user. To this end, we have proposed Embodied Spatial Programming as a new programming paradigm suited for learning computational thinking concepts. A vast body of work related to this idea has been reviewed where we have analyzed theories and projects developed in the past. Based on this thorough review we have designed a set of principles that define Embodied Spatial Programming as an improved framework to learn computational thinking.

With Embodied Spatial Programming, spatial thinking and computational thinking bound together in order to improve the engagement and the learning outcomes when understanding certain programming concepts. We leverage a child's naturally developed spatial mental models in order to generate new constructs that are drawn from existing ones. The understanding of the environment and the use of physical manipulation make the interaction more intuitive and inherent to the human's natural skills. When applying Embodied Spatial Programming, we also take advantage of the flexibility of the digital medium to generate reflective and interactive tools that evolve in real-time with the user and the environment. Guided by these principles, we have seen how Augmented Reality is a well-suited technology that allows us to apply all these concepts in a natural way. After explaining these principles in detail, we have posed an initial foundation for Embodied Spatial Programming as a programming paradigm to learn computational thinking.

The second research goal of this thesis was to develop and validate this new interaction model with an AR authoring platform. We have designed, implemented and evaluated HyperCubes as an application that makes use of physical paper cubes composed in space, with the aim of teaching computational thinking concepts in a creative and playful environment. HyperCubes has been evaluated with a pilot study and a user study. Both studies have proven the high levels of engagement and interest that the application generates in children ranging from 8 to 14 years old. The studies have also shown how the concept of Embodied Spatial Programming and HyperCubes hold great promise as a potential learning framework and platform to learn not only about computational thinking but also about other subjects of the K-12 curriculum such as Physics or Mathematics.

We have been able to successfully implement and deploy the application in a non-lab setting, where children from different ages have been able to play with it. The qualitative user studies performed verify the engagement levels and potential learning outcomes. In order to validate these learning outcomes, a quantitative user study should be performed. The scope and timing of this thesis has only allowed for the qualitative user studies. The application is now on a stable version and ready to be used for a quantitative user study in order to validate these learning outcomes.

The qualitative user studies have demonstrated how children are engaged with the application and are interested in playing with it more than once. The fact that we are using a fairly new technology that they are not used to interact with could grab their attention at the beginning only. We have seen how their engagement levels endure over consecutive days with the same children in qualitative user studies. HyperCubes has also witnessed the interest of teachers and mentors. Those teaching have shown enthusiasm and curiosity towards the application and have seen the potential of the application in being used in the classroom for educational activities.

We have also demonstrated how the learning process can be improved by adding playful and creative components. HyperCubes makes use of the “Creative Learning Spiral” that Mitch Resnick proposed, where children should be able to imagine, create, play, share and reflect. Besides aiming to be an educational tool, HyperCubes is an authoring tool, it enables children to design and create using space and physical tokens in AR. Currently, there is not a big number of tools that allow for authorship in AR. HyperCubes gives the child the opportunity to create in 3D space and making use of her direct physical environment.

It is also important to highlight how the DIY aspect of the application and the use of paper and physical cubes. Keeping the materials low-cost, easily available and as a familiar crafts project, helps in increasing the engagement and involvement of the children. We have seen how they actively want to take part in the building process of the cubes without no one asking them. They mount the cubes and take them home, feeling part of the whole creative environment. To that end, the DIY aspect of the application becomes essential in the creative cycle.



As seen in the background work, there is still plenty of work to be done regarding Computational Thinking in the classroom (Lockwood & Mooney, 2017). Embodied Spatial Programming offers an opportunity to usher new classroom methodologies involving computational thinking and opens the door to authoring in Augmented Reality in an intuitive and natural manner.

## 6 FUTURE WORK

The first step forward in order to validate the application and the interaction model would be to perform a quantitative evaluation in order to assess learning outcomes and quality. For this quantitative evaluation we should design some educational activities to learn about programming and compare the efficiency of HyperCubes to the efficiency of a basic on-screen based application, such as Scratch (Resnick, et al., 2009), Alice (Cooper, Dann, & Pausch, 2000) or a traditional programming classroom methodology. With this quantitative study we would be able to assert the educational nature of the application and the learning rates related to HyperCubes.

The current version of the application is stable and ready for release to the Apple App Store. The next steps involve releasing it to the Apple App Store and open-sourcing the platform. The current status of the technology hasn't allowed HyperCubes to be implemented on Android as the Vuforia library has just been updated with surface tracking for Android within the past few weeks. HyperCubes will be updated in order to be compatible with this new release for Android and will also be made available at the Google Play Store. Several teachers have shown interest in HyperCubes and open-sourcing the application will give them access to it. The code is up-to-date on GitHub and will be made public once is reviewed.

Sharing and remixing are two features that we would like to incorporate into the application. We would do that by synchronizing the platform between devices in order to have multiple users designing and playing with the same digital content in real-time in a collocated environment. Moreover, a digital platform could be designed in order to upload and download projects and play with them or take them further.

We would like to implement HyperCubes for head-mounted displays (HMDs). This would be in order to analyze whether the interaction improves when having the hands free. HyperCubes was designed in order to be accessible for everyone with a smartphone and paper. One of the reasons why HMDs were not considered is because they are expensive and not accessible to the masses. We would still like to compare the interactions between the smart device version and a potential HMD version using, for example, Hololens or Meta 2.

In order to take HyperCubes further, we would like to design a platform to create custom cubes. The application as it is, has a set of eight types of cubes that have behaviors inherently attached to them. We would like to allow for the generation of different types of cubes and the possibility of programming custom behaviors that the user can attach to each side of the cube. The user would be able to draw patterns for each side of the cube, take a picture of them and upload them to the system in order to make them recognizable. An advanced user could then program each side of the cube with code designed by them. This would make the application way more scalable and allow for way more interactions and commands.

Another of the possible future steps is adding electronics inside cubes. This idea was considered at the beginning of the project but it was discarded as the main idea was to make the application accessible to everyone and easy to set up. Having cubes with electronics inside could offer possibilities such as adding LEDs, conductive sides for the cubes or having screens on the sides of the cubes such as in Display Blocks (Pla & Maes, 2013), a project developed by Pol Pla at MIT Media Lab. Embodied Spatial Programming and HyperCubes could serve as an interface to plenty of projects that make use of electronic-embedded physical objects such as Paper Signals (Blankensmith, SmoothTechnology, & GoogleCreativeLab, 2017) or Chibitronics (Qi & Paradiso, 2015). Using Augmented Reality, the user could program the electronics of each piece and add a layer of digital content to enhance the visual output.

Finally, HyperCubes is a platform where one can add any type of programmable digital content. In this thesis, we have focused on a set of basic constructs that we have found to be essential for learning about programming. New content can be added into the application and different cubes can be created in order to teach different subjects. Open-sourcing the platform is just a means to open the application to infinite possibilities in the educational arena.

## 7 REFERENCES

- Amores, J., Benavides, X., Comín, M., Fusté, A., Pla, P., & Miralles, D. (2012). Smart Avatars: using avatars to interact with objects. *Proceedings of the 3rd IUI Workshop on Interacting with Smart Objects*.
- Antle, A. N. (2009). LIFELONG INTERACTIONS: Embodied child computer interaction: why embodiment matters. *Interactions*, 16 (2), 27-30.
- Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13 (1), 20-29.
- Bers, M. U. (2017). *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom*. New York: Routledge.
- Bers, M. U. (2010). The Tangible Robotics Program: Applied Computational Thinking for Young Children. *Early Childhood Research & Practice*, 12 (2).
- Bewley, W. L., Roberts, T. L., Schroit, D., & Verplank, W. L. (1983). Human factors testing in the design of Xerox's 8010 "Star" office workstation. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 72-77). New York: ACM.
- Blankensmith, I., SmoothTechnology, & GoogleCreativeLab. (2017). Paper Signals. Retrieved from Voice Experiments: <https://papersignals.withgoogle.com/>
- Bush, V. (1945). As we may think. *The atlantic monthly*, 176 (1), pp. 101-108.
- Case, N. (2018, July 20). Explorable Explanations: 4 More Design Patterns. Retrieved June 1, 2018, from Nicky Case's blog: <https://blog.ncase.me/explorable-explanations-4-more-design-patterns/>
- Cooper, S., Dann, W., & Pausch, R. (2000). Alice: a 3-D tool for introductory programming concepts. *Journal of Computing Sciences in Colleges*, 15 (5), 107-116.
- Ellis, T. O., Heafner, J. F., & Sibley, W. L. (1969). *The GRAIL Project: An experiment in man-machine communications*. Santa Monica: RAND Corporation.
- Englebart, D. C. (1986). The Augmented Knowledge workshop. *Proceedings of the ACM Conference on The history of personal workstations*. Palo Alto: ACM.
- Fernandez-Baena, Adso, & Miralles, D. (2014). AvatAR: Tangible interaction and augmented reality in character animation. *Proceedings of the 3rd IUI Workshop on Interacting with Smart Objects*, (pp. 28-31).
- Fry, B., & Reas, C. (2003). Processing. Retrieved from Processing: <https://processing.org/>

Fusté, A., Amores, J., & GoogleCreativeLab. (2017, August 28). Paper Cubes. Retrieved from Google Experiments: <http://experiments.withgoogle.com/ar/paper-cubes>

Good, J., Romero, P., du Boulay, B., Reid, H., Howland, K., & Robertson, J. (January, 2008). An embodied interface for teaching computational thinking. Proceedings of the 13th international conference on Intelligent user interfaces (pp. 333-336). Gran Canaria, Spain: ACM.

Goyal, S., Vijay, R. S., Monga, C., & Pratul, K. (2016). Code Bits: An Inexpensive Tangible Computational Thinking Toolkit For K-12 Curriculum. Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction (pp. 441-447). Eindhoven, Netherlands: ACM.

Heun, V., Hobin, J., & Maes, P. (2013). Reality editor: programming smarter objects. Adjunct Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication (pp. 307-310). Zurich, Switzerland: ACM.

Heun, V., Stern-Rodriguez, E., Teyssier, M., & Maes, P. (2016). Reality Editor. Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (pp. 4-4). San Jose, California, USA: ACM.

Hideyuki, S., & Kato, H. (1993). AlgoBlock: a Tangible Programming Language - a Tool for Collaborative Learning. Proceedings of 4th European Logo Conference (pp. 297-303). Athens, Greece.

Horn, M. S. (2006). Tangible programming with quetzal: Opportunities for education. Boston: Tufts University.

Horn, M. S., & Jacob, R. J. (2007). Tangible programming in the classroom with tern. Extended abstracts on Human factors in computing systems (pp. 1965-1970). New York: ACM.

Ishii, H., & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. Proceedings of the ACM SIGCHI Conference on Human factors in computing systems (pp. 234-241). New York: ACM.

Jordà, S., Geiger, G., Alonso, M., & Kaltenbrunner, M. (2007). The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces. Proceedings of the 1st international conference on Tangible and embedded interaction (pp. 139-146). New York: ACM.

Kafai, Y. B., & Peppler, K. A. (2011). Youth, technology, and DIY: Developing participatory competencies in creative media production. *Review of research in education*, 35, pp. 89-119.

Kay, A. (1972). A Personal Computer for Children of All Ages. Proceedings of the ACM National Conference. Boston: ACM.

Kay, A. (1987). Doing with Images Makes Symbols Part 1. Retrieved from Internet Archive: <https://archive.org/details/AlanKeyD1987>

Kay, A. (1987). Doing with Images Makes Symbols Part 2. Retrieved from Internet Archive: [https://archive.org/details/AlanKeyD1987\\_2](https://archive.org/details/AlanKeyD1987_2)

Kay, A. (2005). Squeak Etoys, Children & Learning. Glendale: Viewpoints Research Institute (VPRI).

Kay, A. (1996). The early history of Smalltalk. In T. J. Bergin Jr, & R. G. Gibson Jr., History of programming languages---II (pp. 511-598). New York: ACM.

Klassner, F., & Anderson, S. D. (2003). Lego MindStorms: Not just for K-12 anymore. IEEE Robotics & Automation Magazine, 10 (2), pp. 12-18.

Lockwood, J., & Mooney, A. (2017, March 22). Computational Thinking in Education: Where does it Fit? A systematic literary review. Ithaca, NY, NY.

Magenat, S., Morderchai, B.-A., Klinger, S., & Sumner, R. W. (2015). Enhancing robot programming with visual feedback and augmented reality. Conference on Innovation and Technology in Computer Science Education (pp. 153-158). New York: ACM.

McNerney, T. S. (2000). Tangible programming bricks: An approach to making programming accessible to everyone. Massachusetts Institute of Technology, Dept. of Architecture. Program In Media Arts and Sciences. Boston: Massachusetts Institute of Technology.

Newcombe, N. (2017, August 25). The connection between spatial intelligence and STEM. (M. Kim, Interviewer)

Nielsen, M. (2017, September). Working Notes on ChalkTalk. Retrieved June 1, 2018, from Cognitive Medium: <http://cognitivemedium.com/interfaces-1/index.html>

Papert, S. (1987). Constructionism: A New Opportunity for Elementary Science Education. National Science Foundation, DRL Division Of Research On Learning In Formal and Informal Settings. Cambridge: Massachusetts Institute of Technology.

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

Peng, H. (2012). Algo. Rhythm: computational thinking through tangible music device. Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction. (pp. 401-402). Kingston: ACM.

Perlin, K., & Nunes, G. B. (2017). Atypical: A Type System for Live Performances. Adjunct Publication of the 30th Annual ACM Symposium on User Interface Software and Technology (pp. 61-62). New York: ACM.

Piaget, J. (1971). The theory of stages in cognitive development. New York: McGraw-Hill.

Pla, P., & Maes, P. (2013). Display blocks: a set of cubic displays for tangible, multi-perspective data exploration. Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction. New York: ACM.

Pokress, S. C., & Dominguez Veiga, J. J. (2013, October 7). MIT App Inventor: Enabling personal mobile computing. arXiv:1310.2830 .

Qi, J., & Paradiso, J. (2015). Crafting technology with circuit stickers. Proceedings of the 14th International Conference on Interaction Design and Children. ACM.

Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. *Journal of Computing Sciences in Colleges*, 25 (5), 66-71.

Radu, I., & MacIntyre, B. (2009). Augmented-reality scratch: a children's authoring environment for augmented-reality experiences. Proceedings of the 8th International Conference on Interaction Design and Children (pp. 210-213). Como, Italy: ACM.

Repenning, A. (1993). Agentsheets: a tool for building domain-oriented visual programming environments. Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems (pp. 142-143). Amsterdam: ACM.

Resnick, M. (2017). *Lifelong kindergarten*. Boston: MIT Press.

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

Rosenbaum, E. (2010, September). Color Code. Retrieved June 1, 2018, from Eric Rosenbaum's personal website: <https://www.ericrosenbaum.com/color-code/>

Schneider, B., Jermann, P., Zufferey, G., & Dillenbourg, P. (2011). Benefits of a Tangible Interface for Collaborative Learning and Interaction. *IEEE Transactions on Learning Technologies*, 4 (3), 222 - 232.

Skulmowski, A., Pradel, S., Kühnert, T., Brunnett, G., & Daniel Rey, G. (2016, January-February). Embodied learning using a tangible user interface: The effects of haptic perception and selective pointing on a spatial learning task. *Computers & Education*, 64-75.

Soleimani, A., Green, K. E., Herro, D., & Walker, I. D. (2016). A Tangible, Story-Construction Process Employing Spatial, Computational-Thinking. Proceedings of the The 15th International Conference on Interaction Design and Children (pp. 157-166). Manchester, United Kingdom: ACM.

Sutherland, I. E. (1968). A head-mounted three dimensional display. Proceedings of AFIPS, (pp. 757-764).

Sutherland, I. E. (1964). Sketch pad a man-machine graphical communication system. Proceedings of the SHARE design automation workshop (pp. 6329 - 6346). New York: ACM.

Victor, B. (2011, November 8). A brief rant on the future of interactive design. Retrieved September 1, 2017, from Bret Victor's personal website: <http://worrydream.com/ABriefRantOnTheFutureOfInteractionDesign/>

Victor, B. (2017, December). DynamicLand. Retrieved June 1, 2018, from DynamicLand: <https://dynamicland.org>

Victor, B. (2011, March 10). Explorable Explanations. Retrieved June 01, 2018, from Bret Victor's personal website: <http://worrydream.com/ExplorableExplanations/>

Victor, B. (2012, September). Learnable programming. Retrieved June 1, 2018, from Bret Victor's personal website: <http://worrydream.com/#!/LearnableProgramming>

Victor, B., Schachman, T., Te, P., Horowitz, J., & Iannini, L. (2016). Realtalk. Retrieved from Human Advancement Research Community: <https://harc.ycr.org/project/realtalk/>

WWW group. (1998). www. Retrieved from www: <https://www.org/>

Wai, J., Lubinski, D., & Benbow, C. P. (2009). Spatial ability for STEM domains: Aligning over 50 years of cumulative psychological knowledge solidifies its importance. *Journal of Educational Psychology*, 101 (4), 817-835.

Weller, M. P., Yi-Luen Do, E., & D. Gross, M. (2008). Escape machine: teaching computational thinking with a tangible state machine game. *Proceedings of the 7th international conference on Interaction design and children* (pp. 282-289). Chicago, Illinois: ACM.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49 (3), 33-35.

Yang, H. S. (2015). Variable reality: interacting with the virtual book. Massachusetts Institute of Technology, Department of Architecture. Program in Media Arts and Sciences. Boston: Massachusetts Institute of Technology.

Zünd, F., Ryffel, M., Magnenat, S., Marra, A., Nitti, M., Kapadia, M., et al. (2015). Augmented creativity: bridging the real and virtual worlds to enhance creative play. *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, 21.



## 8 TABLE OF FIGURES

Figure 1. From left to right, Seymour Papert with the Logo Turtle	16
Figure 2. Creative Learning Spiral by Mitchel Resnick	18
Figure 3. Projects, Passion, Peers and Play by Mitchel Resnick	19
Figure 4. An example of a Logo problem to draw a flower	21
Figure 5. Computational thinking skills and approaches	23
Figure 6. Scratch Interface	24
Figure 7. Computational thinking pedagogical platforms	24
Figure 8. SmallTalk being used on an Alto computer, by children	27
Figure 9. ChalkTalk sketch example	28
Figure 10. GRAIL Interface (1966)	30
Figure 11. Doing with Images makes Symbols	31
Figure 12. MetaDesk platform (Ishii & Ullmer, 1997)	34
Figure 13. Children using AlgoBlock system (Hideyuki & Kato, 1993)	35
Figure 14. Tern and Quetzal programming interfaces	36
Figure 15. ARScratch, Color Code and CodeBits Interfaces	38
Figure 16. DynamicLand (Victor, 2017)	39
Figure 17. Reality Editor	40
Figure 18. Enhancing Robot Programming with Visual Feedback and Augmented Reality	41
Figure 19. Chinese Abacus	42
Figure 20. Modern Abacus (20 <sup>th</sup> Century)	42
Figure 21. Ivan E. Sutherland demonstrating Sketchpad	43
Figure 22. The first Graphical User Interface on the Xerox Star 8010	44
Figure 23. Wall-E movie depicting a future with inactive humans surrounded by 2D screens	45
Figure 24. Child playing with well pump. Ten fingers, both hands.	46
Figure 25. Child playing with touchscreen. One finger, one hand	46
Figure 26. Hololens, Meta and Magic Leap One Head Mounted Displays	48

Figure 27. Regular glasses	49
Figure 28. Meron Gribetz interacting with Meta headset	50
Figure 29. Hololens user interacting with Hololens system	50
Figure 30. Kid playing with blocks	51
Figure 31. Gestures we are used to perform	53
Figure 32. Turning a page of a book	53
Figure 33. The Sword of Damocles, 1968	55
Figure 34. Programming examples vs. Embodied Spatial Programming	59
Figure 35. The Logo Turtle (Papert, 1980)	60
Figure 36. Children making markers for HyperCubes (Leicester Art-AI Festival 2018)	62
Figure 37. Paper cubes made by 10-year-old students (Leicester Art-AI Festival)	63
Figure 38. Paper Cubes Google AR Experiment (Fusté, Amores, & GoogleCreativeLab, 2017)	65
Figure 39. HyperCubes Application Interface	67
Figure 40. Side view of HyperCubes Application Interface	68
Figure 41. Emitter cube	68
Figure 42. Transform cube	70
Figure 43. Split cube	71
Figure 44. Logic cube	73
Figure 45. Physics cube	74
Figure 46. Effects cube	76
Figure 47. Gaming cube	77
Figure 48. Hyper cube	78
Figure 49. HyperCubes: Emitting one sphere	80
Figure 50. HyperCubes: Example with an Emitter cube and a Transform cube emitting people	81
Figure 51. HyperCubes: Aerial perspective. Split and Transform cubes	82
Figure 52. HyperCubes: Aerial perspective. Scene explained	82
Figure 53. HyperCubes: Side perspective Split and Transform cubes	83

Figure 54. HyperCubes: Conditions Menu on Checkpoint behavior	84
Figure 55. HyperCubes: Gaming scene being resolved	84
Figure 56. HyperCubes: Configuration of cubes in space	86
Figure 57. HyperCubes: User creates song with Sound behaviors	87
Figure 58. HyperCubes paper markers	88
Figure 59. From left to right, pattern used for the design of the cubes and features (yellow marks) recognized by the AR algorithm	90
Figure 60. Color palette selected for the set of cubes	91
Figure 61. Cube side example with pattern surrounding glyph (Animal behavior)	91
Figure 62. HyperCubes GUI: Editing Emitter Parameters.	92
Figure 63. HyperCubes GUI: Editing CustomSplit Parameters.	92
Figure 64. HyperCubes Emulator	95
Figure 65. From left to right, Vuforia online application to load Multi-Targets and an Image Target tracking rating	96
Figure 66. HyperCubes: Generating Rockets and splitting them up in the air.	97
Figure 67. HyperCubes: When pointing upwards, the virtual content persists even if the cube are not in view.	97
Figure 68. HyperCubes at the Leicester Art-AI Festival	99
Figure 69. Samples of post-it feedback from teachers and students	99
Figure 70. Tutor Survey Results	100
Figure 71. Relevant responses from the tutor's survey	101
Figure 72. Exerpts from the media interviews to teachers and students at the Leicester Art-AI Festival	102
Figure 73. HyperCubes at the Parts&Crafts Girl's Invention week	104
Figure 74. Children mounting cubes and cubes mounted	109
Figure 75. Children playing with HyperCubes at MIT Museum	110

## 9 APPENDIX

### *Qualitative user study researcher journal*

**Location:** Parts & Crafts, Somerville, MA, USA

**Date:** 25<sup>th</sup> / 26<sup>th</sup> July 2018

**Context:** Girls Invention Week summer camp

**Users:** 8-14 year-old children.

#### **DAY 1**

The iPad is heavy

The amount of outputs they can create is fairly limited.

More behaviors and parameters would let them create more. Need for more parameters.

Interaction with UI is way better than the first version of the app.

All of them understand the LOOP concept. They mention Scratch and coding.

They are engaged and want to play with it repeatedly.

A group of two try to create a song with multiple SOUND behaviors.

They are very impressed and use words like 'cool' or 'awesome' when they see the app working for the first time.

They ask about the different behaviors and what each one of them does. They show a lot of interest.

A lot of them want to make cubes: cut the patterns and mount them. One girl makes a whole set of them.

They debate in groups and try to argue what will happen and what cubes they need to use in order to achieve a certain goal.

They make use of space to make different things happen.

They place PORTALS in different spots on the room. They understand that the boids will disappear and appear on the other side before happening. They make use of space.

They understand very quickly how to use parameters in order to change the variables on each cube.

They understand it quickly because they want to change parameters to achieve certain goals.

They understand computing behaviors and LOGIC behaviors.

A girl tries to use the MODULUS behavior to get three boids forward and one down each time.

Several of them like to create CHECKPOINTS and solve them themselves.

The different models and effects make them very engaged.

Especially the ANIMALS. They go through all of the animal models and try to get specific ones, giving them color and splitting them.

One of the student places a COLOR behavior after a SPLIT. She stops to think why not all of them have color. Then she understands that she has to place it in front of the SPLIT in order to get all of them with the same color. She understands sequencing.

The teacher/monitor is impressed with the app and takes photos.

Some of them use the word 'weird' to describe the output sometimes.

They play with the physical cubes once they have them built. One girl tries to juggle with them.

They try to track cubes in space, holding them with their hands. And they stack them to get to the boids that go up with the balloon.

They try new things like: what happens if we put a bunch of cubes all at once?

They try to think about the output in advance. They use reasoning to project and think about what will happen.

They want to play with the PHYSICS cube although it is not fully implemented.

They have curiosity for PHYSICS and understand the concepts in advance. A girl throws a cube to explain GRAVITY.

When the teacher tells them to clean, they want to keep on playing with the app.

It is the last activity that has kids playing with it.

They don't want to leave.

Several of them ask for the app to take home.

## **DAY 2**

Students that played with HyperCubes the day before come back as soon as I put the cubes out.

They know how to use the app straight away from the day before.

They want to put ANIMALS.

They use the BALLOON to make the ANIMALS go up.

They play with ANIMALS in the air. They hold cubes with their hands in space and they track them.

They make people disappear in one side of the room and appear in another side.

One of the kids makes a song and dances to it.

One of the students wants to make the boids go up and then down. She puts the BALLOON first and all of the boids go up. She puts the DOWN behavior after the UP behavior but the boids are going up so they never reach the DOWN behavior. She then realizes she can use a MODULUS behavior to make one boid go up then another boid go forward and down.

She also stacks the cube up and places them on top of a toy tower in order to split the boids up in the air.

A little girl (8) spends a lot of time playing with the cubes, just tracking them and tweaking parameters. Placing them in different dispositions and trying to see if it works. Trial-error.

Sometimes, when they reach for the cubes while holding the iPad, looking through the camera, they cannot synchronize the hand with the position of the cube. The iPad is too big and the camera has a big offset with reality.

It looks like younger kids (9-12) want to play more time than older kids (12-14).

In this second day with the same students it looks like they are less interested in the app. They don't stay that long to play with it. A lot of them are engaged in other activities happening at the same time such as 'boat making', 'sword making', screen printing, movie making, etc. Although many of the students that played the day before come back to play again. And some of them stay longer than the day before, using more cubes and more behaviors.