Frientelligent: Autonomous multi-agent collaboration, competition, and interaction curriculum for young children

by

Raul Alcantara

S.B. in Electrical Engineering and Computer Science
Massachusetts Institute of Technology (2022)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Authored by: Raul Alcantara
Department of Electrical Engineering and Computer Science
June 2023

Certified by: Cynthia Breazeal
Professor of Media Arts and Sciences at MIT
Thesis Supervisor

Certified by: Sharifa Alghowinem
Research Scientist in Media Arts and Sciences at MIT
Thesis Supervisor

Accepted by: Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Frientelligent: Autonomous multi-agent collaboration, competition, and interaction curriculum for young children

by

Raul Alcantara

Submitted to the Department of Electrical Engineering and Computer Science
on June 2023, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

AI has become a critical part of our day-to-day activities, often operating discretely as a core component within our day-to-day devices. Despite its hidden nature, it's important to teach and learn about it as that would provide transparency about how revolutionary it is, what are their limitations, and how one can leverage its potential responsibly and ethically.

Moreover, introducing AI to students at a young age not only helps them have more understanding and appreciation of this technology, but it gives them a chance to contribute to the community later on. Furthermore, early education in AI can help students develop critical thinking skills, creative problem-solving abilities, and a heightened awareness of the ethical implications surrounding AI.

Unfortunately, there is still a gap in the literature for methods to teach AI to young learners, even when considering the traditional lecture-based style of teaching. To fill this gap, this research creates a novel AI curriculum and pedagogy that delivers AI concepts related to multi-agent interaction to students of ages 9-14. We evaluate the effectiveness of our curriculum in learning AI concepts, and in keeping students engaged throughout.

In the curriculum, where multi-agent collaborate or compete, we teach the concepts of path planning and policy making. We do this by having students use a web-based interface where they'll be able to control the different policies a robot can take and see how this makes a difference on their behavior. Students were given the opportunity to try two different versions of the same game: A *virtual* version, where all the interaction happens on a computer; and a *physical* version, in which they have to rearrange physical bots, "obstacles" and "rewards" (made of Legos) to build their own playfield.

To evaluate the effectiveness of our curriculum, we gave students a pre and post-questionnaire to see how their knowledge of different AI concepts had changed. To evaluate how engaging students found the interface, we took observation notes, had a post-interview where we asked them about their experience, and recorded their interaction to look for signs of excitement/boredom.

For the *virtual* mode, we found that the majority of students enjoyed the freedom the UI gave them to construct and manipulate their desired elements while witnessing the real-time replication of their actions on their friend's screen. However, a few of them were overwhelmed by the multitude of available options. Likewise, for the *physical* mode, students really enjoyed being able to physically interact with different objects, see their changes detected in the interface, and observe the corresponding movements of the robots following the policies they had selected. Similarly, we noticed that overall students' knowledge and confidence about the AI concepts increased after performing our activity.

We conclude that using a collaborative web-based interface can be a useful way to teach AI concepts and that even though using physical objects for learning makes the experience more engaging, an all-virtual approach gives students more freedom for quick trial-and-error, which increases their learning as well.

There are several ways in which this work can be extended. For instance, introducing customization options for each bot, such as allowing students to give them unique names, could greatly enhance student engagement. Additionally, expanding the library of themes and assets available would provide more opportunities for children to build according to their preferences and desires. By addressing these aspects, we can further enrich the learning experience and make a bigger impact on the student's learning and perception of AI.

Thesis Supervisor: Cynthia Breazeal
Title: Professor of Media Arts and Sciences at MIT

Thesis Supervisor: Sharifa Alghowinem
Title: Research Scientist in Media Arts and Sciences at MIT

Acknowledgments

I want to first thank my supervisor, Sharifa Alghowinem, for the countless days of continuous help, encouragement, and just overall sweetness (both figuratively and literally). Similarly, I want to thank Randi Williams for her immense help in making this research a... real research, and also just for being a great friend. Thank you, Tasneem Burghleh for helping me make with the development just when I needed help. The website wouldn't have been the same without you. Thank you, Deem Alfozan for your help making the videos, those were such a nice highlight of the full experience. I want to thank the 4 of you for helping me run the studies. I really would be nowhere without all of you! Thank you Cynthia Breazeal for the good insight about the project and thank all the awesome people from PRG, I'm really *really* lucky to have been beside you all :').

Segunda parte, obviamente a mi familia. No hay palabras que puedan describir lo orgulloso y feliz que estoy de ser parte de la familia Alcántara Castillo. Ma, Marivel Castillo, gracias por ayudarme desde el principio y seguir siendo mi mayor fan. Pa, Raúl Alcántara Romero, gracias por haber hecho tanto por mi y por todos nosotros para llegar a donde estamos. Kchets, Angie Alcántara, creo que todo lo puedo decir aqui tu ya lo sabes (o al menos eso espero). El ser tu hermano ha sido (y esta siendo) una de las experiencias más divertidas que he tenido. No tienes idea lo orgulloso que estoy de ti. Estoy bastante felices de que puedan venir a esta graduación. Los amo infinitamente y los estoy esperando aqui <3

No puedo hablar de familia sin agradecer a ustedes, Mama Olga, Papa Lalo y Mama Delia por ser unos abuelos grandiosos. De la misma manera, gracias Papa Alfredo, me hubiese encantado haberte conocido pero siempre te llevo en mi corazón. Esto va para todos ustedes! Gracias a todos de ambas familias por todo :D

Yizhi Wang, also known as Cherry, also known as Beb. Thank you, for being there for me for more than 4.5 years, and counting. I'm not kidding when I say that if I owe surviving MIT to anyone, I owe it to you. I may be leaving MIT, but I'm glad I'm still going with you. Qué más puedo decir excepto Wǒ ài nǐ

Antes que me olvide, gracias Sr. Raul y Sra. Elena, ustedes me ayudaron cuando mas lo necesitaba y siempre les estare agradecido.

Last but not least, I want to thank my friends. Shinjini, someone who I knew I could always trust and that has been with me through hard times (including classes XD). Gyalpo, I've been working with you for a couple of months and something tells me that might increase; It's great to have you as a friend. More thanks to the amazing friends from MIT y Peru: Sohini, Hector, Vik, Muhender, Oscar, Marcelo, Sagnik, Preeti, Gustavo, Luz, Rodrigo, Manuel, Steven, Diana, Emmy, Evelyn, Chanwoo, Andrew. If you think you should be on this list, you probably should I just have very bad memory :(

Quien hubiese creído que alguien de Barranca, Lima, Perú estaría obteniendo una Maestría del MIT. Ni yo me lo creo, pero al perecer es cierto. Me emociona pensar lo que el futuro depara!

In summary, thanks everyone. I couldn't have done this alone. Here goes to more good stuff.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As AI becomes more popular and relevant in our day-to-day activities and in our society in general, there is an increasing need for the new generation of students to know about AI's core concepts. Moreover, given that currently AI is sometimes hidden into our day-to-day devices, teaching students about it would provide transparency about how revolutionary it is, as well as what are their limitations. Finally, this would empower them with the tools necessary to understand how (and why) the current AI technologies work.

When it comes to teaching AI, the most popular method is the traditional lecture-based style, given how convenient it is to deliver it, and because that has been the default way of teaching for centuries. However, lecture-based learning takes away the existent interaction students have with the real world, which can hinder engagement and thus the effectiveness of learning. Instead, more interactive lessons (ones that can bridge the effects of the real world into relevant concepts from the subject learned) can be much more effective in delivering the core concepts in a way that it's much easier to digest. AI lends very well to that methodology given how widespread AI is on our society.

A common approach for teaching in an interactive manner is the use of games, due to its inherent engaging nature. Most of these attempts utilize a fully-digital approach, because they don't require as much hardware (usually a phone or computer is enough). However, evidence suggests that a "learning by doing" approach where students interact with tangible objects can be more effective [16].

1.1 Research Questions

This research investigates the effectiveness of a novel AI curriculum and pedagogy that delivers AI concepts related to multi-agent interaction. This is performed by investigating the following sub-questions.

1. **Curriculum on autonomous navigation**: We investigate the usefulness of developing an AI curriculum focused on autonomous navigation, a topic that has not been built upon very much in the past.

2. **Interface Design**: We investigate how useful can a web-based UI be to improve the engagement of learning kids

3. **Education evaluation**: We investigate good questions to ask before students perform our activity and then after, with the hopes that it'll be a good indicator of whether the student learned part, if not all, of the curriculum presented to them.

1.2 Contributions

To summarize, this thesis makes the following contributions:

1. Developing an AI multi-agent interaction curriculum that teaches about agents collaboration and competition using concepts such as path planning and policy making, which has not been the focus of previous curricula.

2. Implemented a web-based interface that students can use to control the behavior of the robot (either virtual or tangible).

3. Published the web-based interface used in this research, as well as its curriculum, so that it can be used by any instructor interested in teaching these concepts around the world.

Finally, it's worth noting that due to the lack in the research literature for robot-based learning, the replication of any related work (or this one) can still be valuable.

1.3 Thesis Outline

In Chapter 2 we will introduce the research's background and related work. In Chapter 3 we'll go into details about the methodology in our study, in particular about our UI, Curriculum and how we measure success. In Chapter 4 we'll go into a in-depth description about the technologies behind the UI and our study. In Chapter 5 we'll show the results of our study and up to what extent users found the UI useful for learning. Finally, in Chapter 6 we'll discuss our conclusion and future work.

Chapter 2

Background and Related work

2.1 Background

2.1.1 AI education

AI has improved significantly in the past 50 years. More impressively, AI is now embedded into our day-to-day life more than ever before [6], and we interact with it constantly in more than one way (e.g., Google Maps navigation, virtual assistants like Alexa or Siri, etc). This constant exposure to AI sparks curiosity on young students to figure out how these technologies work [10]. Moreover, the rapid evolution of AI is changing the dynamics of our society [11], requiring the new generation a different type of skill set than it did years ago. As a result, it is on our best interest to prepare kids to be familiar with AI and how it works.

It's also worth noting that the development of AI curriculum has shown effectiveness on teaching the core concepts of AI as well as making them aware of the use and importance of AI. [4] [11] [17] [21]

2.1.2 Digital games for Education

For the purpose of this thesis proposal, *digital games* refer to the ones that do not interact with objects from the "real world", and are instead fully contained in a virtual environment. These games usually are only played on one device at a time, like a cellphone, tablet or computer, and do not require interaction with the surrounding area nor much physical movements from the player. Some type of games that would *not* fit this criteria would be board games or sport simulators (e.g., golf, tennis).

When it comes to game-based learning, most of them are digital [14]. Moreover, digital games can reach a bigger audience because they do not require any additional hardware other than what most people already have (i.e., cellphones, computers, etc). As a result, digital games have been found to be useful to transmit information about several topics, in either the literature [5] [1], as well as in the mainstream world (e,g., Kahoot [20], MinecraftEdu [3]).

2.1.3 Physical games for Education

Physical games are the counterpart of *digital games*. They refer to the activities that require some type of interaction between the player and their surroundings. Note that there can still be a digital component (e.g., a social robot that require students to use a tablet to communicate with it), as long as not everything is digital.

Physical games have the potential of being more engaging than the digital ones due to their haptic interactive nature. This makes them a great tool for learning and has already been used to teach different subjects like Math [15], Computer Science [13], Physical Education [19] and many more.

2.1.4 Tangible objects for learning

The use of tangible objects for learning is thought to be a good way of bridging the real world and abstract concepts, which can ease the transition to learning a new technology like programming, AI, or anything CS-related [7]. In the context of learning, a very common example of tangible objects are the usage of robots as learning companions, teaching assistants or simply as a source of learning materials [12]. For example, [21] uses a cellphone's screen and different Lego pieces to create a robot companion, which they found was helpful for delivering the material in a fun way.

2.2 Related work

In Table 2.1 we present different works that are focused on teaching people the fundamentals of AI using non-conventional (i.e., not lecture based) methods while maintaining a good level of engagement with their students. 5 out of the 8 that are showed here are from [8], which is a review paper of the literature in AI Education. That paper focuses on 17 different projects, and we only picked the ones that are relevant to this project. In general, we did not include projects that were too generic in what they were trying to teach. For example, we excluded several of them that focused on teaching Computer Science (or programming) rather than AI in particular.

Paper	AI Focus	Game Description	Age group
[18]	Machine Learning, Neural Networks, Decision Trees	Game was agriculture-based. The main purpose was to classify mangos into 1) sweet, not sweet and 2) good quality, medium quality, bad quality. Students used an UI that allowed them to drag and drop blocks to construct flowcharts of the ML training.	Middle school students
[9]	Fuzzy decision making, Evolutionary computation, Neural Networks	The game was a battlefield, where each student received "sensor data" about their surroundings (comprised of other student's and different obstacles), and use that information to fire "photon blasts". Students were expected to code logic that allowed them to blast as many opponents as possible, while not losing all their energy. Students competed against other students, as well as instructor-made characters.	Undergraduate students

[2]	Decision Making, Heuristics	In the game two teams compete with the purpose of sinking each other's ships. Students have information about the field, but not about what moves the other team has done (e.g. precise positions, torpedo hits). They can only estimate this information by analyzing the intensity of the sound their submarines hear (which propagates through the water), which is where AI comes into play. The game was made into a tournament among the different students.	Master's students
[8]	Strategy Making	Game: Human Resource Machine. Players have to automate work by programming the employees of an office environment. On each level, players are given a task and a drag-and-drop interface to create an strategy using a limited set of actions.	No target age
[8]	Neural Network, Decision Tree	Game: While True: Learn() Players take up the role of an ML specialist who uses visual programming to complete clients' projects. Players are given a User Interface to design a workflow for training Machine Learning models.	No target age

[8]	Path finding, Pattern identifying, Supervised Learning	Game: ViPER Players are supposed to program a robot to help it find its way to a beacon while avoiding different obstacles. Players learn how machines learn, and engage with concepts such as algorithms, the testing, and training phase, and identifying patterns in the data	Middle school students
[8]	Supervised Learning, Classification	Game: PopBots Students are given an interface based on Scratch Jr. where they can interact with a real-life robot and read data from sensors to complete simple activities like recognizing healthy and unhealthy food or different types of music	Preschool students
[8]	Decision tree	Game: Gladiabots Adversarial game where players control a robot squad with the purpose of defeating the other team. The user interface allows players to build arbitrarily complex pipelines for decision making.	No target age.

Table 2.1: Table showing previous games used for teaching AI

First of all, we can see that there is a wide age range, varying from preschool students to Master's students, though there doesn't seem to be one targeted to high school students specifically. On the other hand, only one of them (PopBots) included a real-life component for the engagement (the robot), whereas the rest of them focused on a fully-virtual world.

Finally, in terms of the AI topics that they focus on, we note that about half of them focus in Machine Learning ([18], [8] {While True: Learn(), ViPer, PopBots}), and the other half focus more on having the students/players come up with different strategies given some heuristics.

2.3 Summary

AI is becoming more and more important for young students as it's becoming more embedded in their daily lives, which makes the need to teach about it at a young age more apparent. One of the most popular ways to teach about AI is through digital-based games, due to its low installation cost. However, learning using tangible objects has shown to be very engaging to students, which in turn helps with how well they can learn the subject in question.

The AI aspects of Autonomous Navigation has not been the focus on previous educational studies, which is why we decided to build upon this topic. Similarly, as digital games for education are known to be engaging/fun, we are taking good care of developing a web-based UI that is pleasing to the eye and reduces the friction of interacting with it as much as possible, while still delivering on the content needed to be learned. In the next chapter, more detail about the curriculum, interface design and evaluation are provided.

Chapter 3

Methodology

3.1 Game and Scenario design

Our game will live in a 2D grid (either physical or virtual) where two robots will either need to compete or collaborate for a common purpose. Apart from the robots, there will be *obstacles* to give constraints to the paths the robots can take, and *rewards* to offer students an incentive to choose some paths over others. We hope that these objects will help students reflect on the importance of *path planning* and *policy/strategy making*.

In the *virtual* mode, students will be able to drag and drop the different objects into the grid. In the *physical* mode, students will be able to move their Doodlebot and also the obstacles/rewards (which are made of Lego pieces), and the system will detect it through an overhead camera. In both scenarios, students have the freedom of designing their own obstacle courses.

Apart from designing their playground, students will have the chance to decide their bot's policies and how they should move. This is the part where they will apply the knowledge they have learned thus far.

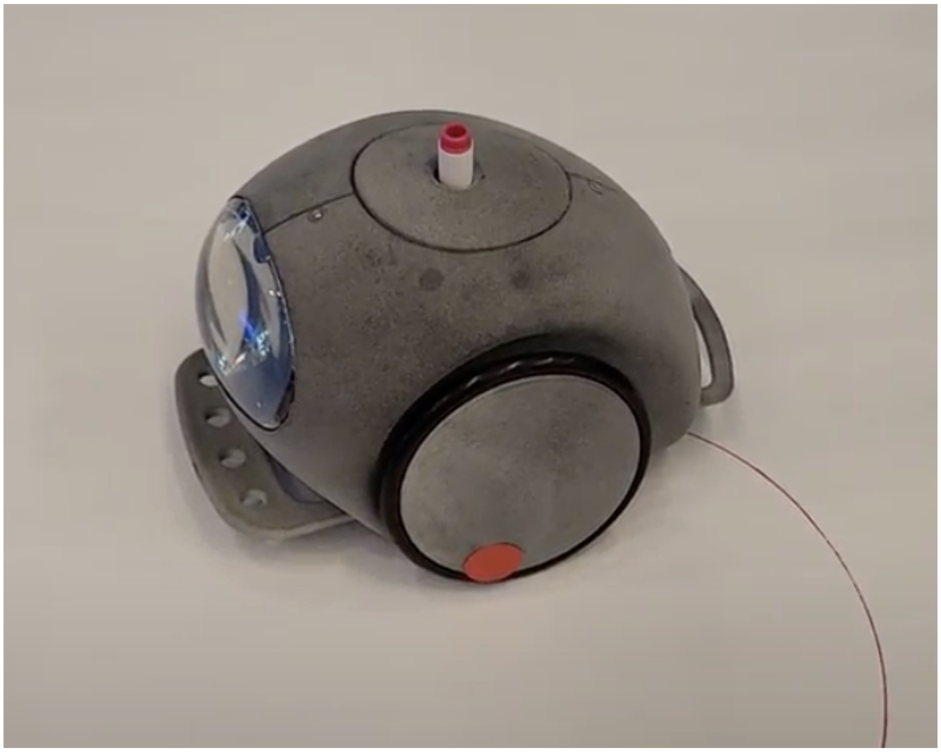


Figure 3-1: Picture of a doodlebot with a pen for drawing

3.2 Doodlebot

The Doodlebot (Fig 3-1) is a robot created by the MIT Media Lab's Personal Robots Group. Its purpose is to engage students in programming the robot to create drawings, and in doing so facilitate their understanding of Machine Learning concepts. For the purposes of this study, we used the bot's API to 1) move it a given distance and 2) rotate it by a custom angle. The bot was able to connect to our web-based UI using BLE technology and a UART messaging protocol.

3.3 Interface

One of the main features of our interface is its synchronization among different users. This means that to go from one page to the other they both need to press 'Continue', or to start the game each of them will have to press 'Start'. To make sure students don't get confused about these capabilities, the UI shows a modal with a helpful message whenever they need to wait for other user's actions, as seen in Fig 3-2

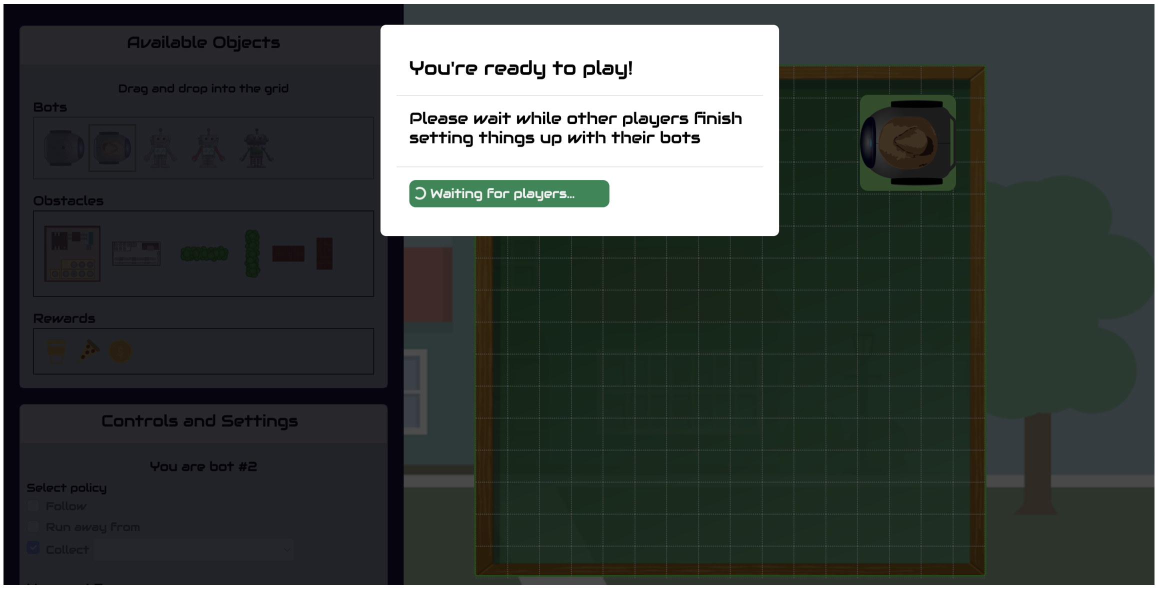


Figure 3-2: UI showing a waiting message

Similarly, the interface itself gives students the flexibility to create scenarios in which they can collaborate toward a common goal (e.g., having one pickup coins and the other one pickup pizza, Fig 3-3). On the other hand, the UI also allows creating adversarial scenarios, like seeing who gets the most coins (Fig 3-4)

Finally, in both versions, there will be 3 users, a teacher that creates a room/connects to the camera, and two students that are the ones deciding what object goes where and what policies to give their bots.

3.4 Curriculum

3.4.1 Tutorial

As previously mentioned, we developed an AI curriculum focused on Automatic Navigation. Students will learn about it through a tutorial that will help them understand 3 different distances: Euclidean, Manhattan, and Dijkstra. For each of them, they watched an explanatory video (created by us, e.g. Fig 3-5), and then see a predetermined version of the activity that will illustrate the strengths (and weaknesses) of using such distances to find how to get from one place to the other (Fig 3-6). These

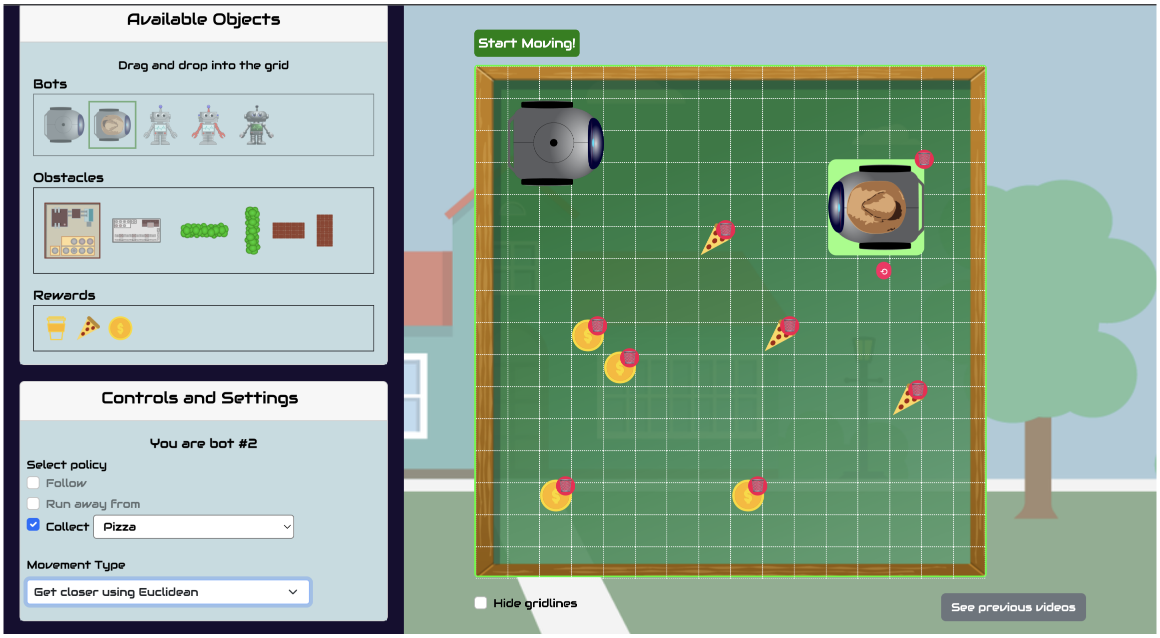


Figure 3-3: Collaboration aspect of the UI to collect different rewards

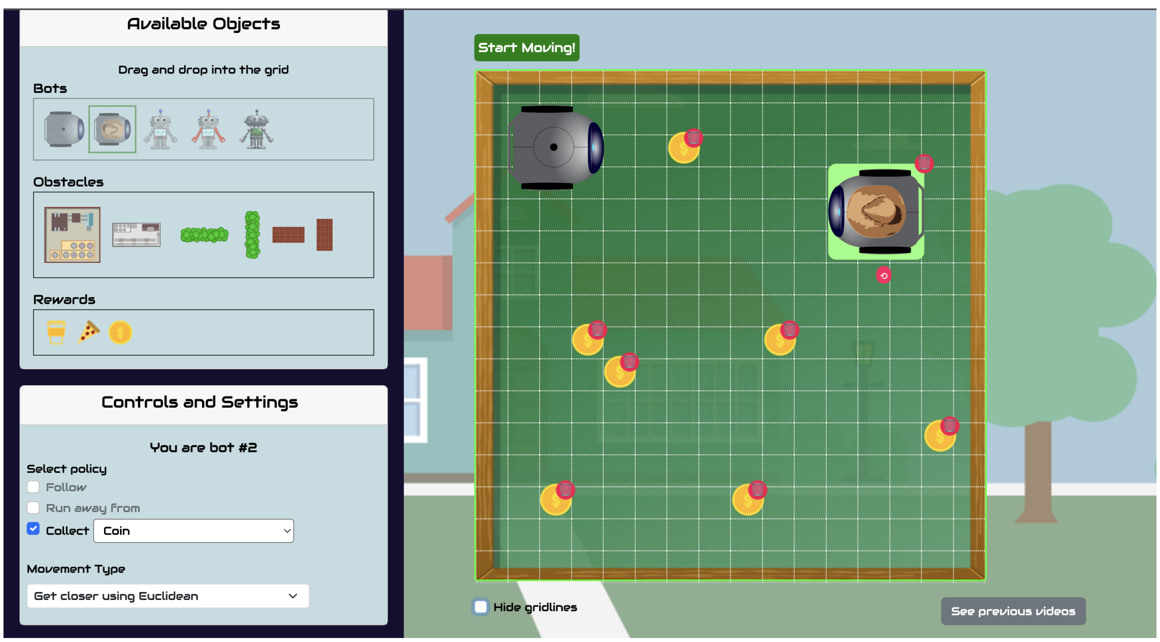


Figure 3-4: Competition aspect of the UI to who gets the most coins

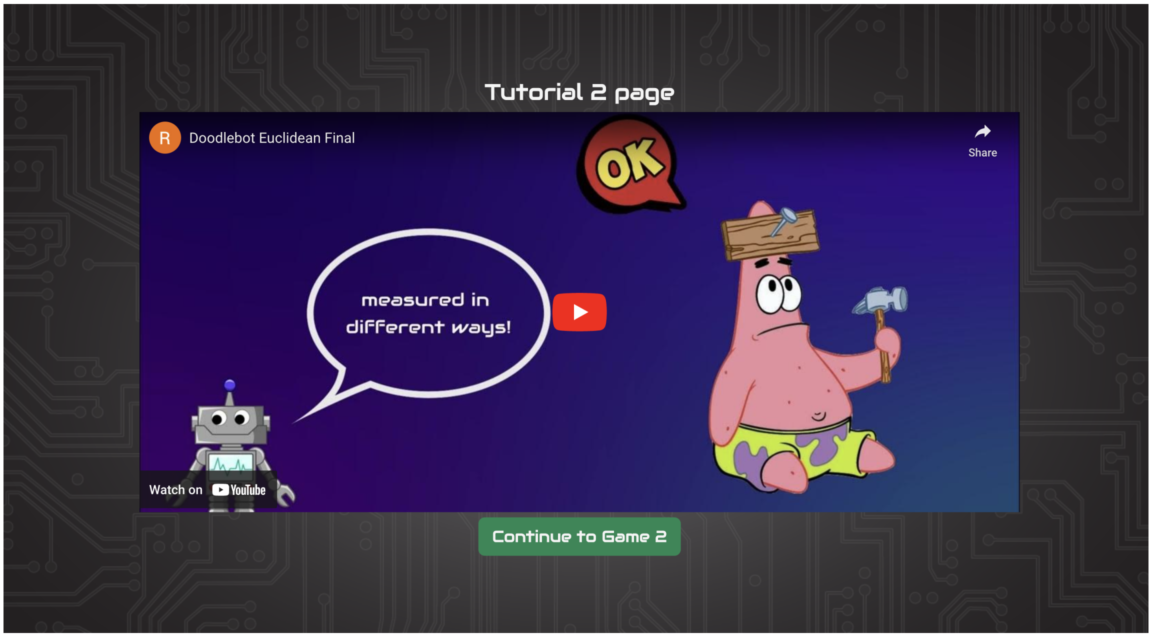


Figure 3-5: Tutorial: Showing a video teaching about the Euclidean distance

activities look like the final game but the only thing they can do here is to press 'Start Game' (as opposed to being able to drag and drop objects into the grid).

Overall, the full tutorial interaction has the following structure:

Type	Description	Purpose
Tutorial 1	Students will watch the Random video	Introduction to the concept of autonomous driving. It poses the question of how long will it take a bot to reach its goal if it moves randomly every time.
Game 1	Students will see the interface for the first time and be able to press the 'Start Moving' button	To have students become familiar with what the interface looks like, to see why moving randomly is not the best solution, and that "we need to be smarter about this".

Tutorial 2	Students will watch the Euclidean video	Video will suggest that one way to go about this is to go to the place that leaves you the nearest. However, for this we need to know how to calculate the distance. The video will then explain the Euclidean distance.
Game 2	Students will see the interface using the Euclidean distance.	To explain what the Euclidean distance is and to show students how using it makes it relatively easy for the bots to pickup the rewards.
Tutorial 3	Students will watch the Manhattan video	Introduction to the Manhattan distance, saying how similar (and different) it is to the Euclidean. Finalize by saying that both distances do not take into consideration any obstacles there might be in between, and posing the question of whether this will be a problem.
Game 3	Students will see the interface with the Manhattan distance	To show students how having obstacles could be problematic if we use a distance that do not take into account obstacles, as the bots will get stuck and not collect their items.
Tutorial 4	Students will watch the Dijkstra video	A high-level explanation of how the Dijkstra algorithm is calculated and how it takes obstacles into consideration.
Game 4	Students will see the same game interface from Game 3, but now with the bots set to use Dijkstra distance.	To have students see how the new interface can help the bots overcome obstacles.

Tutorial 5	Students will watch the Interface video	Explains the final interface capabilities (drag and dropping and bot policy selection), so that right after they can play with it.

Table 3.1: Table showing descriptions of the different parts of the tutorial

It's also worth noting that for the 4 games in the tutorial, most of the positions of the bots/obstacle/rewards remain the same, with only changes when needed: From Random to Euclidean they're the same to show how Euclidean fixes a previous problem. From Euclidean to Manhattan it's the same but adds obstacles to show how the distances are not ready for that, and from Manhattan to Dijkstra they're the same to show how Dijkstra can solve a previous problem. We designed each level with different themes to showcase what they looked like, as for the final activity they'd need to choose a theme. An illustration of what the playing field looks like for Games 1-4 can be seen in Fig. 3-7

3.4.2 Final game/project

For the final activity (either the fully-digital version or the physical one), students will need to find ways of taking their robots between different places while avoiding several obstacles (and other people's robots!), which hopefully will solidify the idea of the steps it takes to perform a path planning task.

Policies

Students will be given full freedom to choose one of three policies "Follow X", "Run away from X", or "Collect Y", where X is a Bot's id, and Y is a type of reward (coin, coffee, pizza, etc). Possible values of Y depend on the theme the students chose. This will change the trajectory that their bot will follow.

Moreover, students will have the chance to choose different movement types: "Random", "Get closer using Euclidean", "Get closer using Manhattan", and "Get

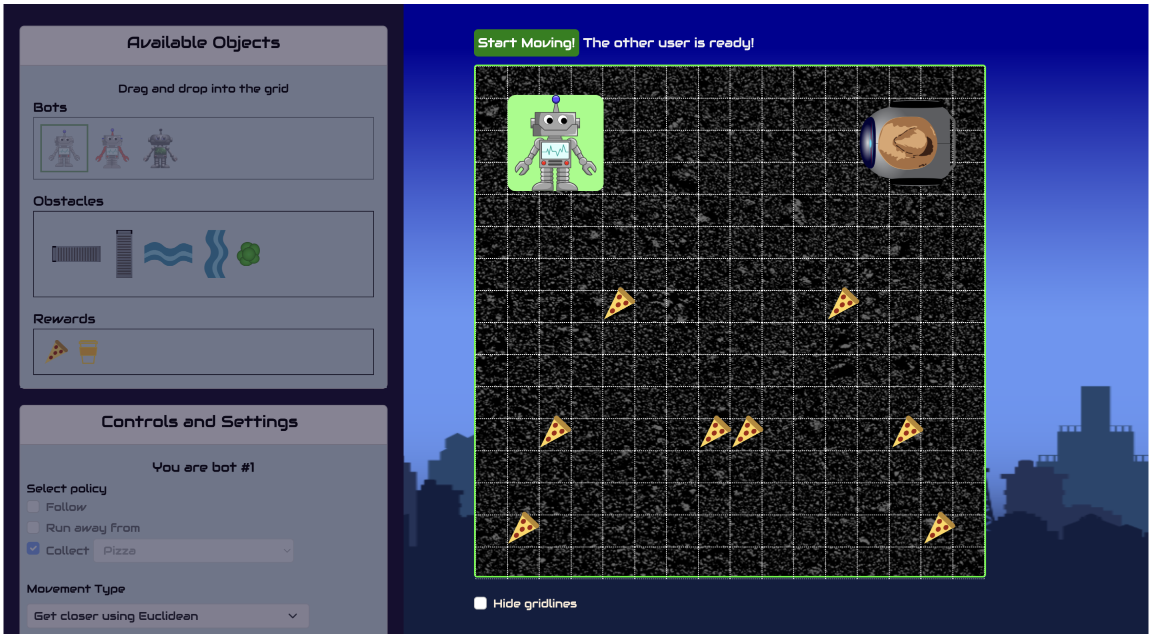


Figure 3-6: Tutorial: Game that is supposed to show the strengths of using an Euclidean distance

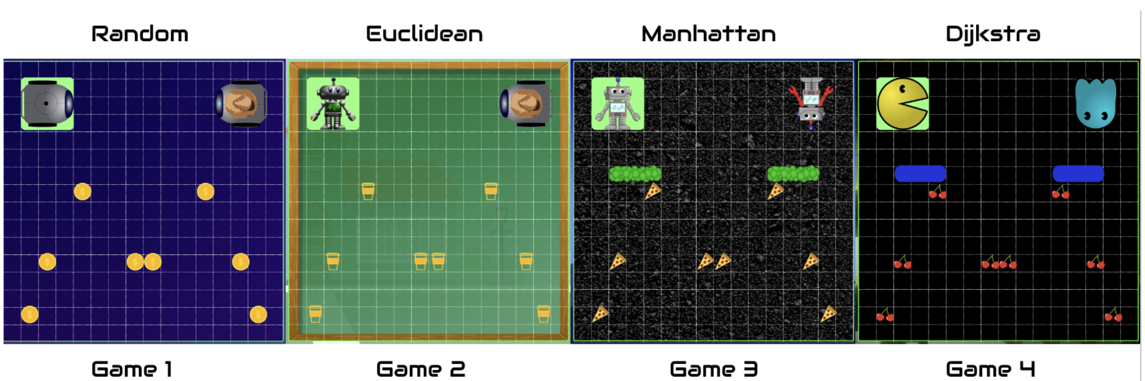


Figure 3-7: Progression of the different games in the tutorial

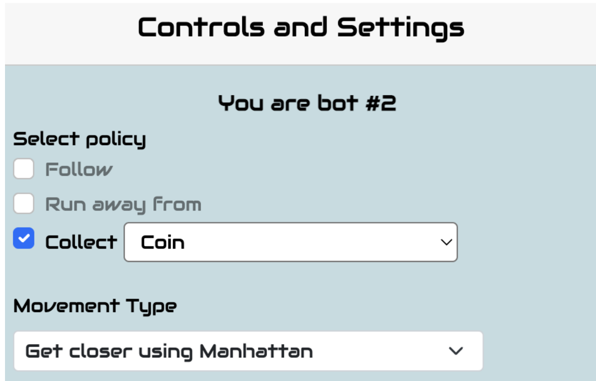


Figure 3-8: Controls of the bots

closer using Dijkstra". This will dictate how the bots will calculate distances and are meant to replicate the 4 tutorial games. An image of how this selection looks like can be seen in Fig. 3-8

Finally, we hope the final activity shows the importance of *Policy/Strategy Making* in AI. Given that students need to choose a specific policy for their robot depending on what they want them to do, this would show them how useful, and even necessary, it is to think well about the strategies they want their AI to have.

Physical mode

For the Physical mode, the final game looks almost identical to the virtual mode, with the difference that the grid background will be a flattened video stream from the overhead camera. An example of this can be seen in Fig. 3-9. IN the actual playground, students will be able to move the bots, obstacle and rewards to create what they want (Fig. 3-10)

3.5 Study Design

3.5.1 Conditions

Even though all students will experience both the virtual and physical mode, we will separate them in two groups (Groups A and B) where the first one will have the *virtual* mode as their first activity, and Group B will start with *physical* instead. We

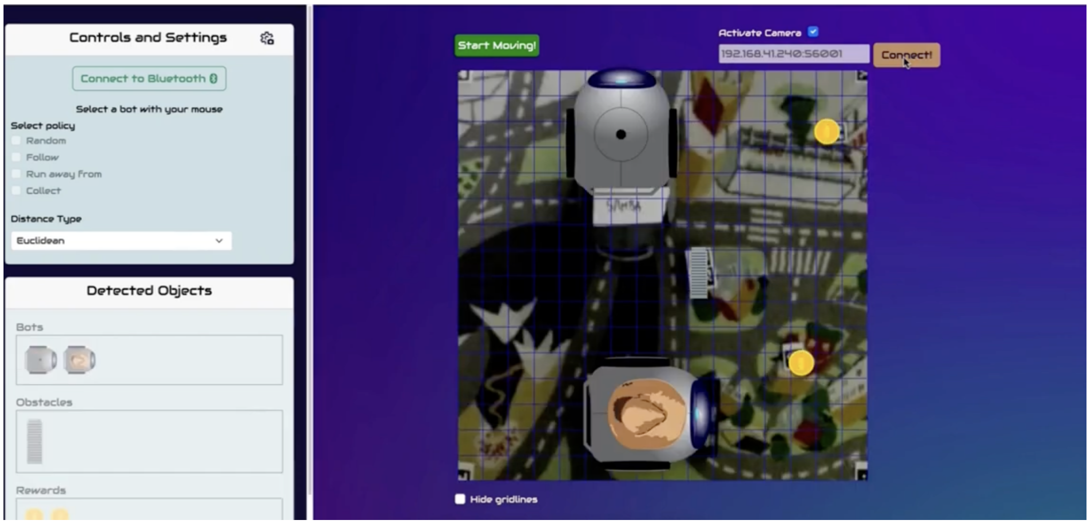


Figure 3-9: UI for camera mode


Figure 3-10: View of the playground. Each object is uniquely assigned an Aruco code, which is placed on the top of its body (for the robots) and on one of the Lego pieces' squares (for the rest).

do this to see if the change in the order leads to different levels of understanding or engagement. This would give us a better idea of how to best present information to kids to maximize learning.

3.5.2 Measures and Data Collection

Pre and post questionnaire

For both groups, we want to make sure that they learned the proposed curriculum. With that in mind, we had a questionnaire before the activity where we asked students knowledge-based questions to gather how much experience they had with the material beforehand. After the first final project (virtual and physical condition), we gave them a form asking about their experience with the project to gather an idea of how much they had liked it. This is also repeated after the second project. At the end of study, we gave them a post-questionnaire identical to the pre-questionnaire to gather an idea of how their knowledge had changed thanks to the activity. The full questionnaire can be found in Appendix A.

Post-workshop interview/discussion

We conducted a 15-minute post-workshop interview/discussion with both students to go over the aspects of the sessions they liked/didn't like, what did they enjoy the most, what did they learn and what are some things that they would change for future versions of this workshop. The pool of questions we used can be found in Appendix B.

Observation Sheet

While the students were playing with the UI or having the tutorial, we took observation notes to empirically check their engagement level. This document can be found in Appendix C.

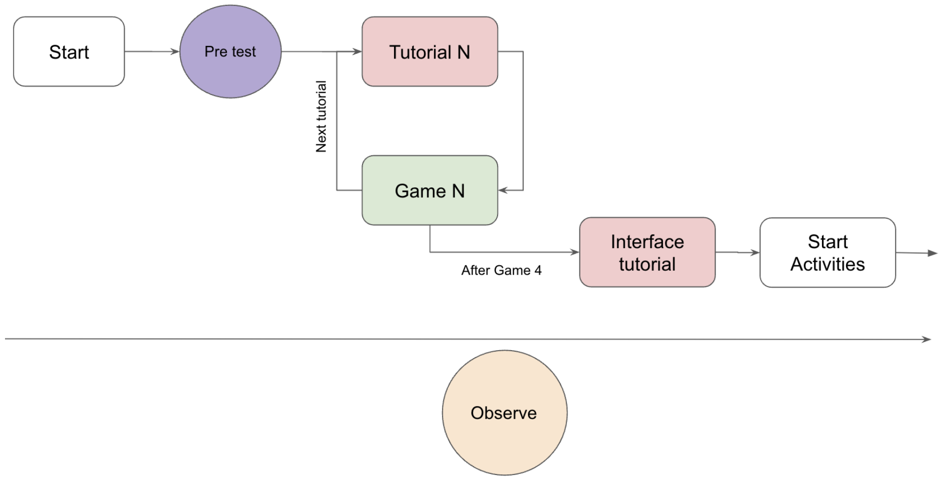


Figure 3-11: Flowchart of the tutorial part. All students will be given a Pre-test first, and then watch different tutorials throughout. While all this is happening, an observer will be taking notes to check for engagement/boredom signs. In this part the kids will be recorded.

Video recordings

We recorded and analyzed the video recordings of the students interacting with the interface to look for signs of engagement/boredom.

Flowchart

In summary, both groups will have the same tutorial, and this flow can be seen in Fig 3-11. After that, their experiences will be similar with the only change that some of them will have *virtual* mode as their Activity #1, and others will start with *physical* instead. This new flow can be seen in Fig 3-12

3.5.3 Participants

In this pilot study, we had 8 students of ages 9-14. For each session, we had two participants teaming with each other. Out of the 8 children, 6 of them were paired with another child, forming 3 different groups, and the other 2 individual kids teamed up with a partner who was not a kid nor part of the study (due to scheduling and cancellations). Every group tried the *virtual* and the *physical* modes, but the order in which they did them changed from group to group. Every group did go through

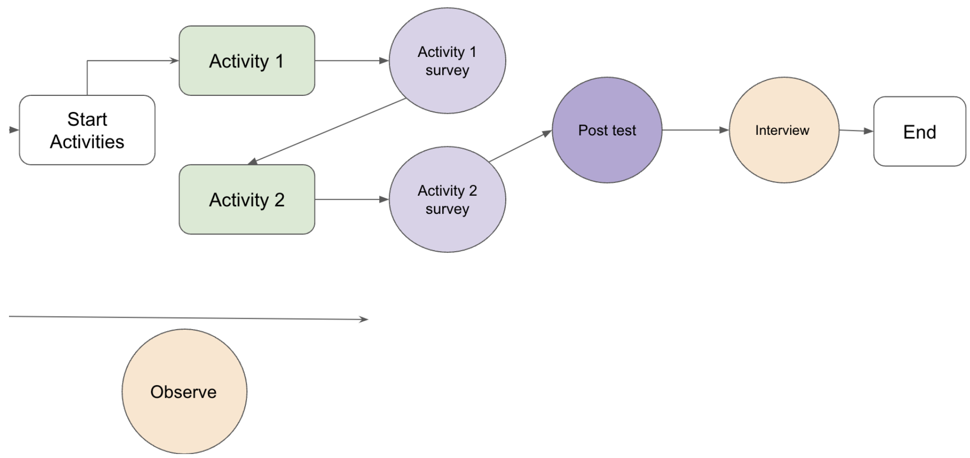


Figure 3-12: Flowchart of the activities. The difference between Group A and B will only change what 'Activity #1' is. Students will be given surveys after each activity, a post-test and an end interview.

the tutorial, which was fully virtual.

-

Chapter 4

Implementation details

4.1 Virtual Grid

For both the *virtual* and *physical* mode, the **VirtualGrid** class was implemented as a way to keep track of all the objects that needed to be shown on the screen. Everything is stored in JSON format. There are 3 types of objects that this class keeps track of: *Bots*, *Obstacles*, and *Coins*. Below we'll explain what each of them are, and what information do they carry. Since our data will be two-dimensional, we'll consider (0, 0) to be the bottom left corner of the grid.

These 3 types of objects carry different information. However, they are all represented as a rectangle in a 2D grid, so they all have the fields found in 4.2

Field	**Description**	**Example**
id	id of object. Should be unique among all the other objects. Often an integer	1
real_bottom_left	[col, row]. Bottom left corner of the object	[1, 2]
width	width of object	3
height	height of object	3
type	Type of object (one of "bot", "obstacle", or "coin")	bot

Table 4.1: Table showing some of the fields that each object stores.

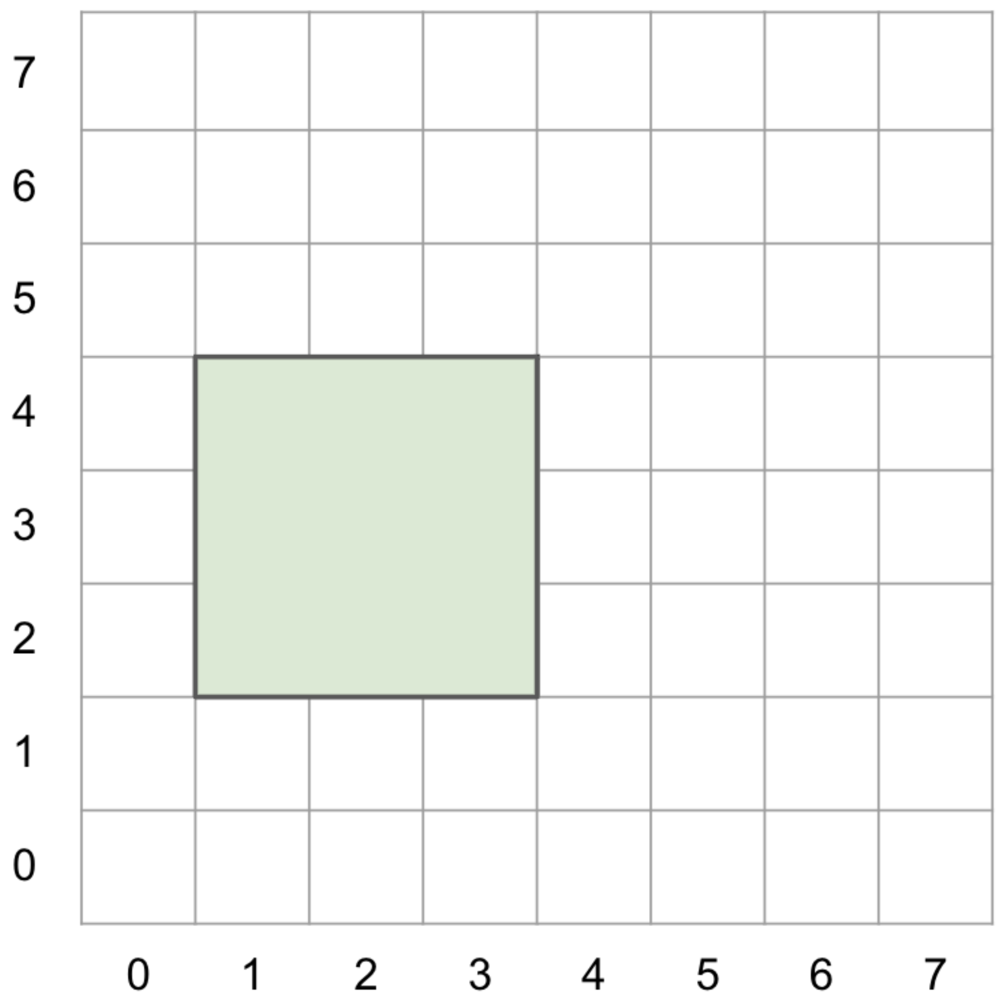


Figure 4-1: Example of an object placement on a grid

For example, the object presented in 4-1 would have a width and height of 3, and it's *real_bottom_left* would be [1, 2]. We note that this information alone is enough to place where (and how big) an object will be.

4.1.1 Bots

Bots are the objects that kids have control over, and the ones that can move.

Field	Description	Example

relative_anchor	It's the "center of gravity" of the bot, and the point that will remain fixed as the bot rotates. On the virtual version, it's often the center, but in the physical version, it depends on where the Aruco codes are placed. It's 'relative' because it's relative to the bottom left of the object, and not of the grid.	[1, 1]
angle	The direction the bot is looking at. It can only be 0 (RIGHT), 90 (UP), 180 (LEFT) or 270 (DOWN)	90

Table 4.2: Table showing some of the fields that each object stores.

4.1.2 Obstacles

Obstacles in principle are objects that bots cannot pass through. These can be city buildings, walls, etc.

4.1.3 Coins

Coins, also known as *Rewards* are objects that Bots are usually trying to collect. Apart from the fields from 4.2, coins have a **coin_collect_type** field that represents the type of coin it is. Examples of this are "Pizza", "Coffee", or "Strawberry". This information is used when kids choose their bot to "Collect X". Here X would be the coin's coin_collect_type.

4.2 Collaboration / Synchronization

As encouraging collaboration is one of the central themes in the study, it was critical to make sure that the infrastructure allowed synchronization between two (and possibly more) users.

4.2.1 Rooms

To enable the best experience for users, we should allow different pairs of users to be able to use the interface simultaneously. For this to be possible, every "game" will be defined in a different "room". This room will be given in the game's url (e.g., *https://....?room=<room_name>*)

4.2.2 Websockets

I used the Javascript library socket.io which allowed them to send event-based messages between the clients and a server. This is a very popular approach for enabling real-time communication.

SocketIO already has the concept of 'rooms', so implementing this logic wasn't a problem. One of the most popular synchronizations we needed to do was to make sure that an activity only happened once the two users had said they were ready to move on. (e.g., moving from a tutorial or starting a game). For all these scenarios, 4-2 is a version of what happens under the hood: Every time a client was ready, it emits an event to the server which would increase the room's counter by 1. If the counter on a room ever reached 2 (i.e., the two kids clicked on it), then the server would emit an event **to everyone in that room** saying they should start.

Allowing this back and forth between clients and the server was possible with relatively few lines of code, as can be seen in 4-3 and 4-4.

Even though coding-wise SocketIO seemed like the better option, this implementation had several flaws, like:

1. Deploying to the web is extremely difficult

2. Need to keep track of both client and server.

3. If the server went down then users of that room info would get erased.

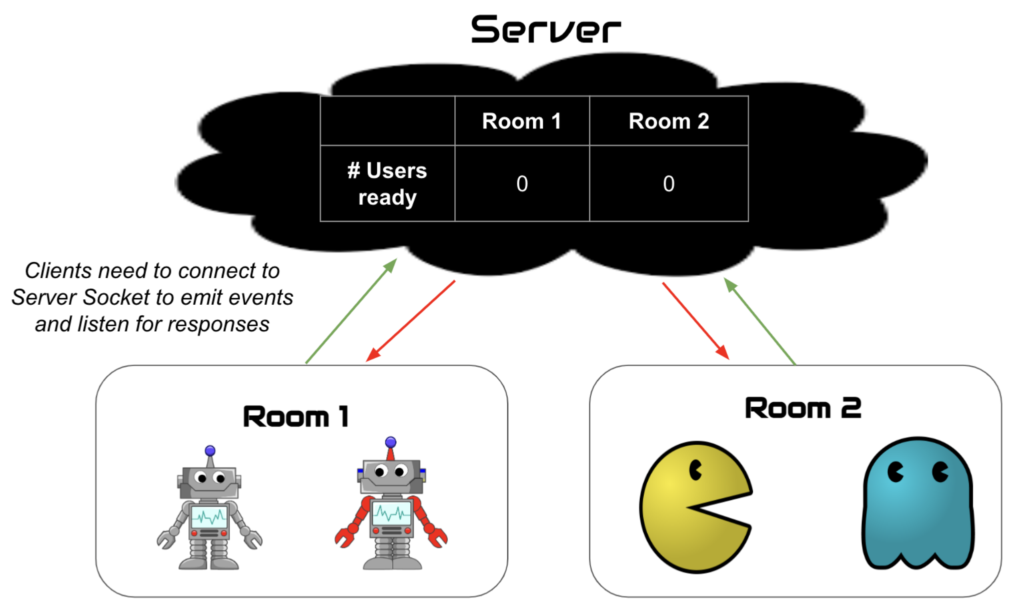


Figure 4-2: Simplification of the architecture using SocketIO

```
//Client code

// When pressing 'Start' button, emit event to server
socket.emit("start", {})

//Wait for server event to start
socket.on("everyone_ready_to_start", () => {
    startGame(); //Actually starts the game for this client
})
```

Figure 4-3: Using SocketIO to synchronize the time of starting game (client).

Solutions to these problems lead to the use of Firebase, which we'll explain in the coming section.

4.2.3 Firebase

With Firebase, this time instead of listening for events, clients would listen to changes from a specific URI. Firebase does not handle 'rooms' natively, but listening for changes in a URI like /**rooms**/<**room_name**> effectively achieves the same result. Firebase allows subscribing to even more specific URIs for a more atomic update

```
//Server code
let rooms = {}; //keeps track of the rooms' info

//Listen for client updates
socket.on("start", () => {
    rooms[socket.activeRoom].users_ready += 1;
    if (rooms[socket.activeRoom].users_ready === 2){
        //Let all other users know the room's ready
        socket.emit("everyone_ready_to_start");
    }
})
```

Figure 4-4: Using SocketIO to synchronize the time of starting game (server).

operation. The new diagram of how this architecture works can be seen in 4-5

A piece of code that achieves the same result as its Socket counterpart would be the one seen in 4-6

Now our *server* would be Firebase but instead of a custom-made server. Given this site is now serverless, it's much easier to deploy (I used Github pages). Now given that all info is stored in Firebase's databases, if a user looses connection or anything of the like, the system will get them where they were prior.

4.3 Interface

4.3.1 Original Design

The first iteration of the interface did not look at all like what it does now. The first iteration looked like 4-7 It's sole purpose was to make sure that I could transform the abstract data from *VirtualGrid* into visual data. For example, the only way in which one could add objects to the scene was to explicitly type where it needed to be, and how big it had to be. Even though this phase of the implementation was good enough for functionality, it was not child-friendly for the study.

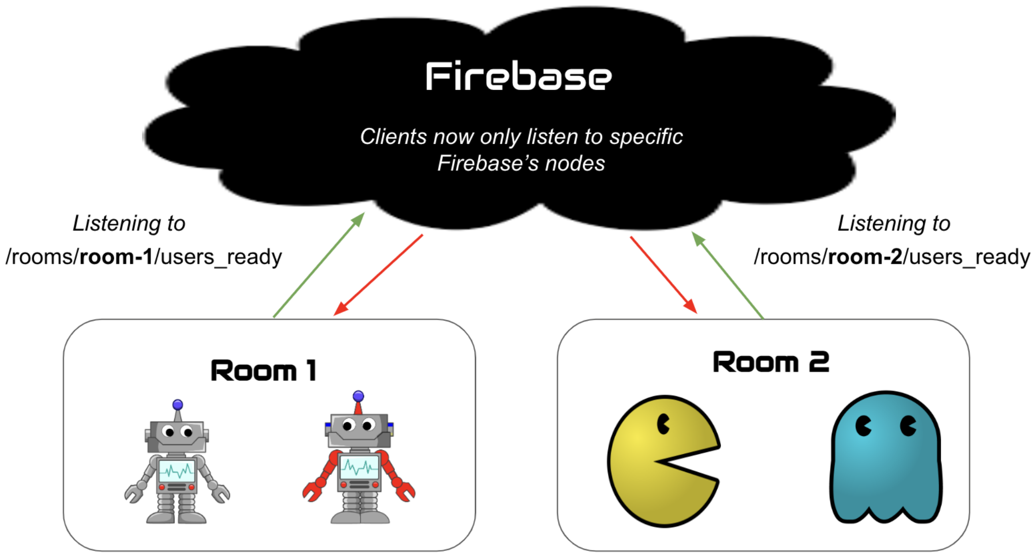


Figure 4-5: Simplification of the architecture using Firebase

```
//Client code

// Listen to given node
let usersDoneRef = ref(db,`rooms/${current_room}/users_ready`);

// When clicked, update the Firebase node
let info = await get(usersDoneRef)
update(usersDoneRef, info + 1)

//Check for updates
onValue(usersDoneRef, (snapshot) => {
    let num_users = snapshot.val() || 0;
    if (num_users === 2){
        startGame();
    }
})
```

Figure 4-6: Using Firebase to synchronize time of starting game.

4.3.2 InteractJS

We knew that if we wanted to make the interface child-friendly, we needed to decrease the amount of typing and clicking. Then it became apparent that allowing drag and drop would be the ideal scenario. After some researching, we decided for using InteractJS for its flexibility and continuous usage from the community.

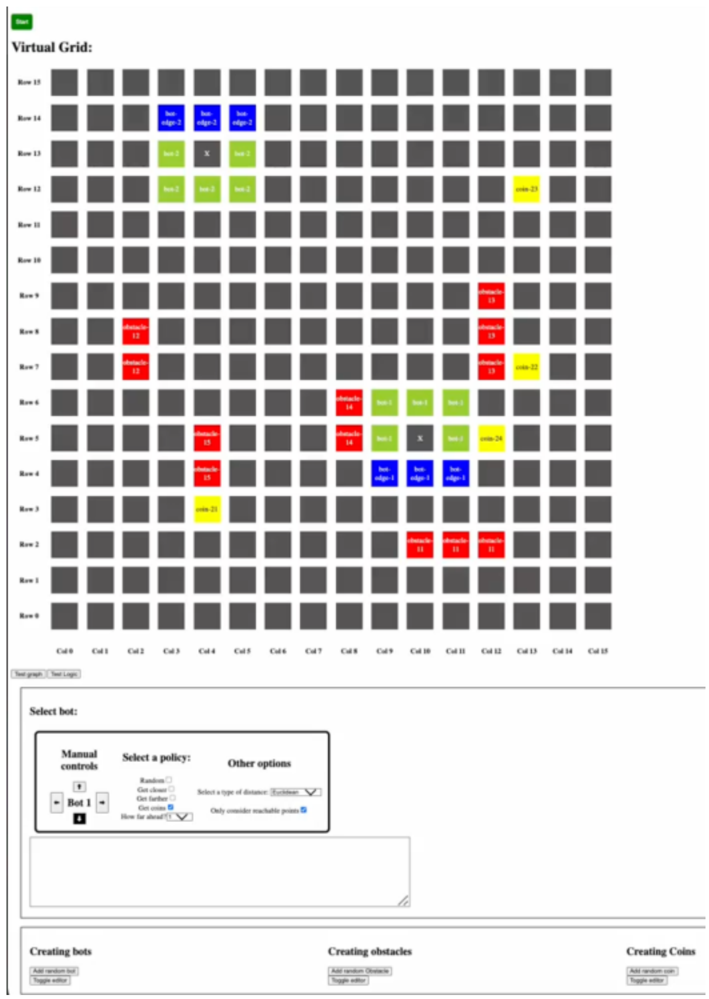


Figure 4-7: Original UI of the Doodlebot Interface. Green/blue objects were bots (the blue side shows the bot looking direction/orientation), Red were obstacles and Yellow were coins.

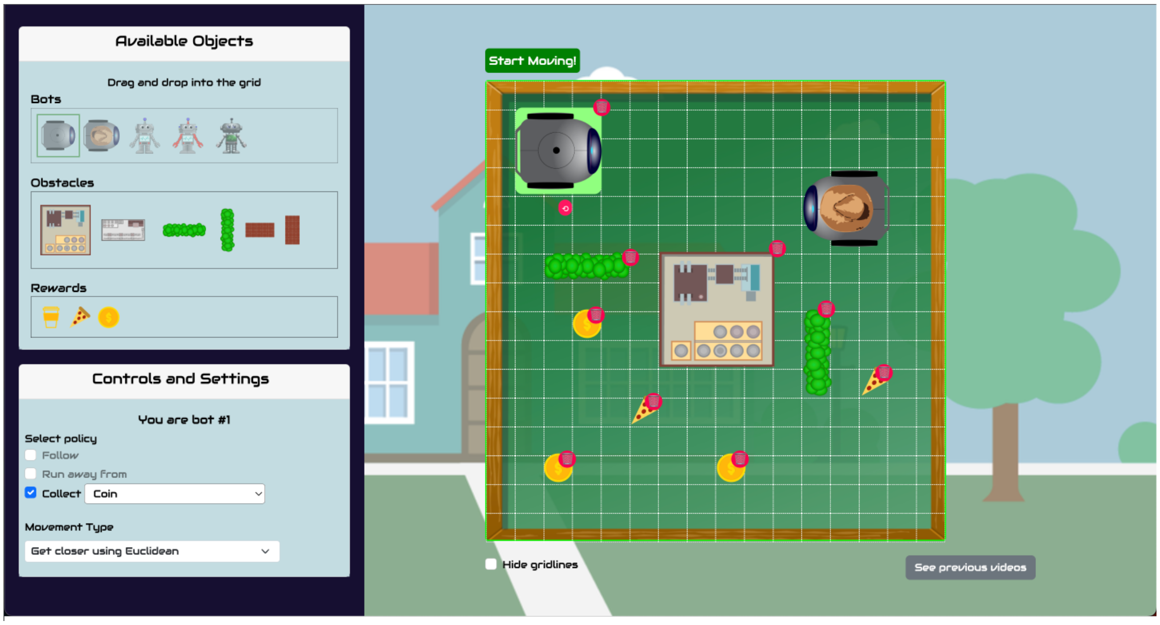


Figure 4-8: New interface, with drag and drop functionality

The way we made this work with our VirtualGrid is that the library on itself gave us screen locations of a draggable object. We knew the screen locations of the grid container, so we could calculate the relative position of a draggable object with respect with the grid. With this and with the knowledge of how many rows and cols the grid had, it just took a bit of math to map the drop position to a VirtualGrid position (i.e., integer coordinates).

With the use of this, and after three weeks of hard work, the latest look of the interface can be seen in 4-8

4.3.3 Bot Options

As seen in 4-9, users are given the choice to 'Select a Policy' and select a 'Movement Type'. Selecting a policy will change the current bot's **bot.policies** field array. This array starts empty and then it will add the values of "Follow", "Run away from" or "Collect". Even though the bot treats it as an array, the current implementation only takes into consideration the highest-ranked one, with the order being Collect > Run away from > Follow. When checking every type of policy, a select dropdown will appear, as these choices are meaningless by themselves. For example, for "Collect"

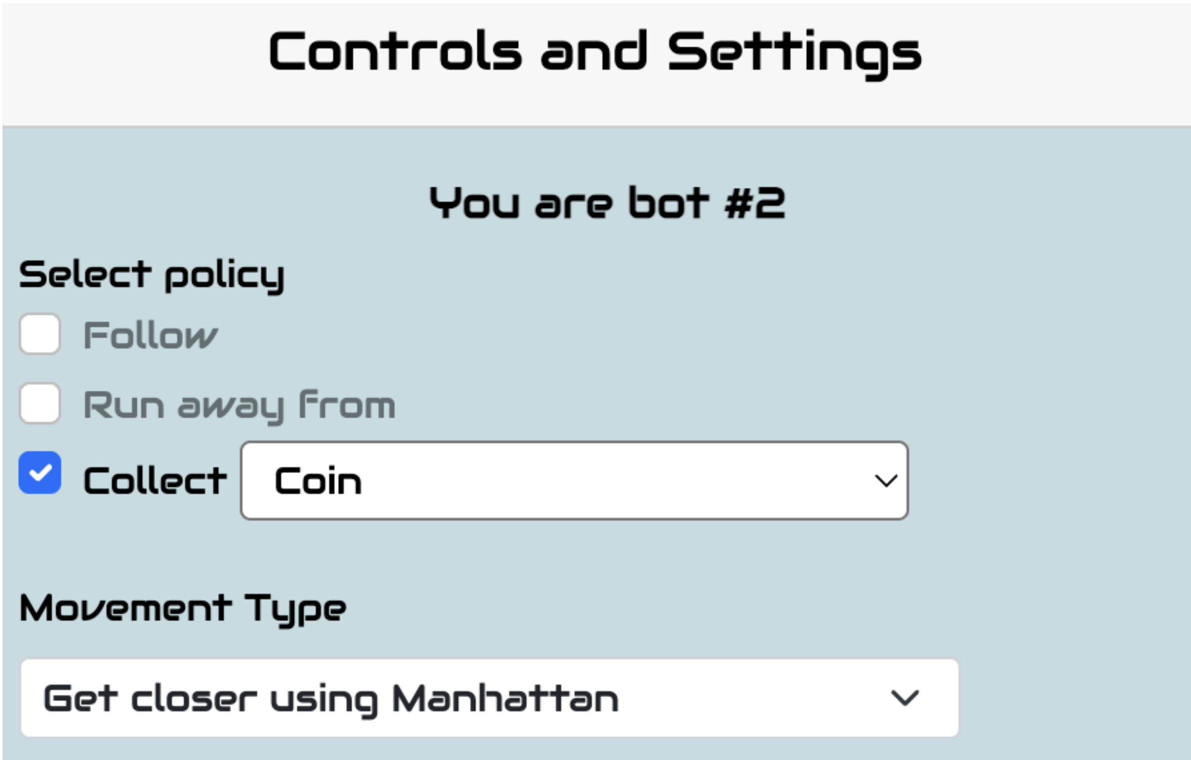


Figure 4-9: Caption

a drop down appears that populates the coins available in the grid, so that a bot can "Collect X". Choosing this value will populate **bot.targets**. Choosing the one for "Run Away from" will populate **bot.run_away_from** and finally choosing the select "follow" will populate **bot.follow**

On the other hand, the Movement Type dictates how the bot will move. We intentionally left this with the 4 options that students learned about in the tutorial. Choosing one of these populates **bot.movement_type**

4.3.4 Themes

The Interface presents 4 themes: None (or default), City, School and Pacman. The difference from one theme to another is the background image and, most importantly, the image assets in the sidebar that are available to drag and drop. The 4 themes can be seen in Fig. 4-10

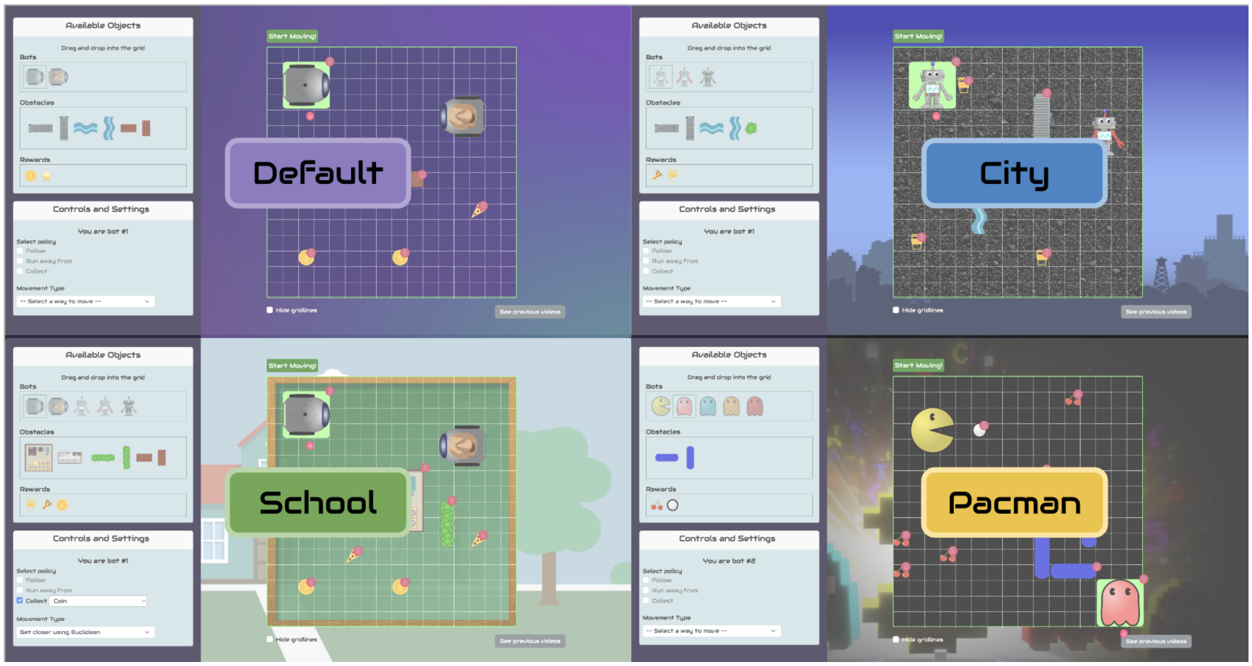


Figure 4-10: All 4 themes with their options

4.4 Tutorial

4.4.1 Setup of rooms

Always one user needs to create a room, and the other one needs to join it. This would start appending ‘?room=<room_name>‘ to every subsequent URL for the Tutorial. The UI in this page can be seen in 4-11 and 4-12. The way we create a room name is by simply choosing a random adjective from a list, and a random noun from another list, and putting them together back to back.

4.4.2 Tutorial and Games

To know when to move from a tutorial to a game, and a game to a tutorial, we use a similar Firebase strategy as the one mentioned in section 4.2.3.

According to the game they are in, the theme will be selected according to our Progression seen in 4-13. Each game will be a pre-defined mode with a given policy type, which will then use to move into the next page.

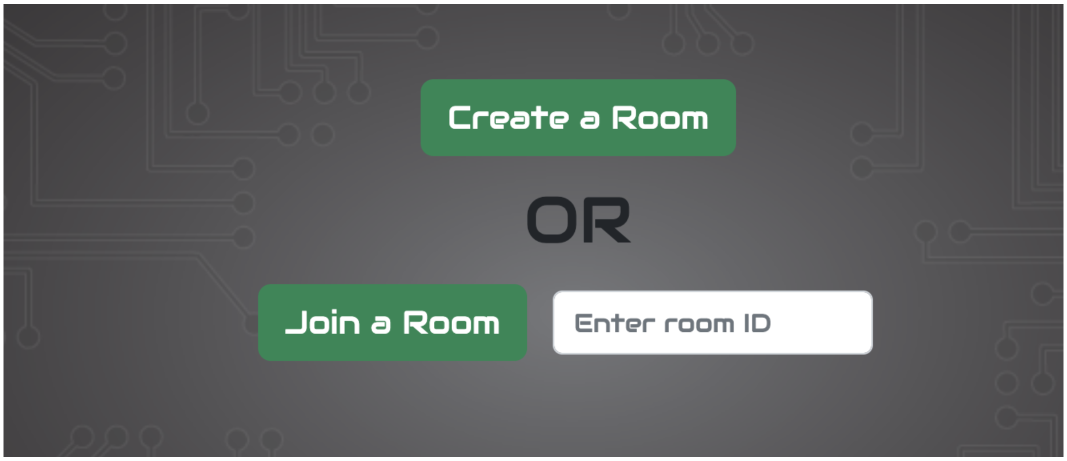


Figure 4-11: Landing page of users. One of them needs to create a room, and the other one need to join them.

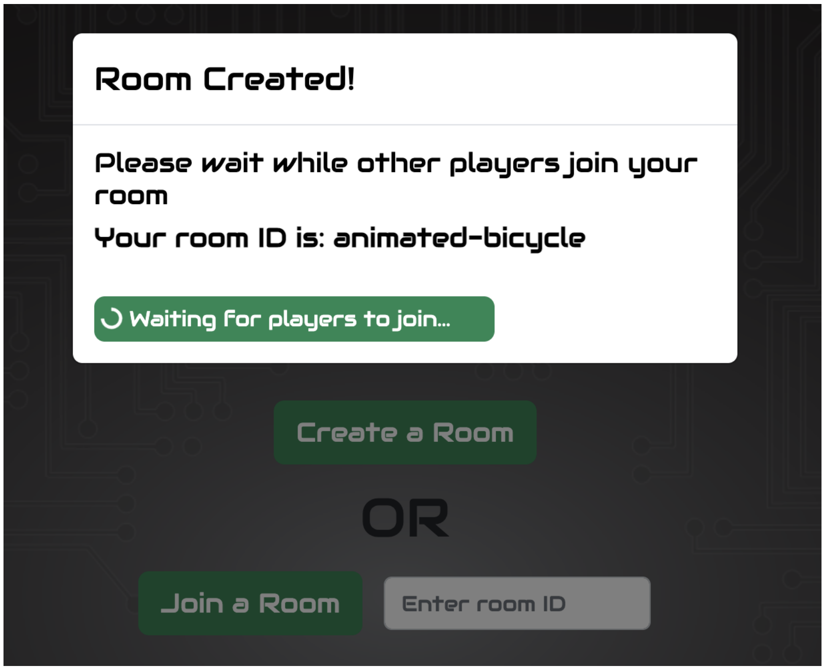


Figure 4-12: Landing page of users. A new room has been created.

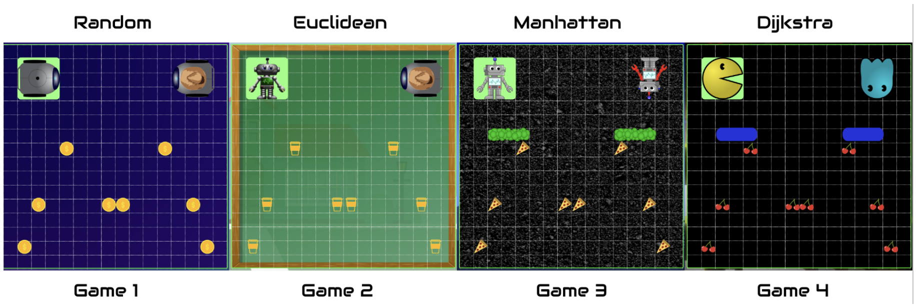


Figure 4-13: Progression of the activities (and themes) of the tutorial.

4.5 Navigation Algorithm

Once the bots start moving, two methods will happen every certain amount of time (assuming that **grid** is the *VirtualGrid* object of this game):

1. **let move = grid.get_next_move_using_policies()** Will use the student's selected policies and movement types to output a move. Possible moves are one of

$$"move", 1$$

,

$$"turn", 90$$

or

$$"turn", -90$$

Which means that the bot can either move forward, turn 90 degrees counterclockwise, or 90 degrees clockwise, respectively.

2. **grid.apply_next_move_to_bot(move)**. Will apply the selected move to the current bot.

Applying a move to the bot is independent of any policy or movement type, so we will only focus on how the **get_next_move_using_policies** method works (see Figure 4-14).

In other words, If the bot's movement type is RANDOM, get a random move and if it's not RANDOM then call the respective method according to the polity the bot has. If any of them returns a valid move, return it. If nothing works, then return a random move.

For either policy (COLLECT, RUN_AWAY_FROM, or FOLLOW), there will be a target of objects (set of coins with type *bot.target*, *bot.run_away_from* and *bot.follow*, respectively) for which we'll want to make the best move to get closer/farther to them. This is not a hard task, assuming we know what 'distance' means.

For this, we have the **distance_to_object(bot, obj)** method, where bot is a

```
function  get_next_move_using_policies(bot_id) {
    let bot = this.bots[bot_id][0];
    let next_move;
    // Random movement takes precedence
    if (bot.movement_type === MOVEMENT_VALUES.RANDOM.value) {
      return this.get_next_move_randomly(bot_id);
    }
    if (bot.policies.includes(BOT_POLICIES.COLLECT.value)) {
      next_move = this.get_next_move_to_collect(bot_id);
      if (next_move) { return next_move; }
    }
    if (bot.policies.includes(BOT_POLICIES.RUN_AWAY_FROM.value)) {
      next_move = this.get_next_move_closer_or_farther(bot_id, false);
      if (next_move) { return next_move; }
    }

    if (bot.policies.includes(BOT_POLICIES.FOLLOW.value)) {
      next_move = this.get_next_move_closer_or_farther(bot_id, true);
      if (next_move) { return next_move; }
    }
    //If nothing works, then just move randomly by default
    return this.get_next_move_randomly(bot_id);
}
```

Figure 4-14: Description of the get_next_move_using_policies method

given Bot and obj can be another Bot, Obstacle, or Coin. A snippet of this method can be found in 4-15

The Euclidean and Manhattan equations are pretty self-explanatory. However, for Dijkstra, there's a bit more to unpack.

4.5.1 Calculating Dijkstra distance

Dijkstra distance is graph-based, so we needed to convert our current grid into a graph, which would allow us to take obstacles into consideration. We created a **GridGraph** class for this. This class takes a Coin as a reference and creates a graph where each vertex represents a possible position of a bot. Every vertex is encoded by its *real_bottom_left* and it's *angle*. For example, for a bot at the origin that is

```
  // ...
  // dx and dy are the deltas of the bot's and obj's centers.
  switch (bot.movement_type) {
    case MOVEMENT_VALUES.EUCLIDEAN.value:
      return Math.sqrt(dx * dx + dy * dy);
    case MOVEMENT_VALUES.MANHATTAN.value:
      return Math.abs(dx) + Math.abs(dy);
    case MOVEMENT_VALUES.DIJKSTRA.value:
      if (obj.type !== COIN_TYPE) {
        return;
      }
      return this.coin_graphs[obj.id].shortest_distance_from_obj(bot);
    default:
      console.log(`Unkown distance_type: ${bot.movement_type}`);
  }
```

Figure 4-15: Description of the distance_to_object method

looking up, its associated vertex would be the string "i->0, j->0, angle->90". The number of vertices is about 4 times the number of cells in the grid (one per each possible angle). Edges are simply valid moves from one position of a bot to another, **assuming you don't pass through obstacles**. Finally, we decided to give each a weight of 1 for simplicity.

Using the Dijkstra algorithm, one is able to populate a minimum distance from the Coin to every other bot position, which would be equivalent to the minimum distance from each bot position to the Coin. Now we're ready to understand **GridGraph.shortest_distance_from_obj(bot)** method in 4-16:

```
  // ...
shortest_distance_from_obj(bot) {
  let { angle, real_bottom_left } = bot;
  let [i, j] = real_bottom_left;
  // Takes 0, 0, 90 to 'i-> 0, j-> 0, angle-> 90'
  let node = this.get_node_from_position(i, j, angle);
  // this.distances was populated using Dijkstra
  return this.distances[node];
}
```

Figure 4-16: Description of the shortest_distance_from_obj method

Given that this is an expensive pre-computation that cannot be done on the fly (like Euclidean or Manhattan), we made it so that if kids choose to use Dijkstra, then they have to click a 'Load bot info' button before starting the game. This button loads this GridGraph for every Coin in the grid and once this is loaded the "Start button" becomes clickable.

4.6 Physical mode

Apart from the challenges of developing the virtual interface, making it to also work with real bots was as demanding, if not more. Here we'll talk about some of the most important aspects of such development.

4.6.1 Connecting to the physical bots

The Doodlebots were developed to communicate through Bluetooth BLE using UART messages. As such, using the browser-available method *navigator.bluetooth* was of much help, as shown in 4-17

```

const device = await navigator.bluetooth.requestDevice({filters: [{
      filters: [{ services: [UartService.uuid] }],
}]})

```

Figure 4-17: Description of how to connect to Doodlebots through BLE

4.6.2 OpencvJs for Aruco Detection

For knowing where every object was, we decided to use Aruco Detection, as we found it easy to use and reliable in terms of identifying unique objects (i.e. each ArUco code has a unique ID, which we use to identify robots, obstacles and rewards). ArUco markers are coded with their relative 3D position, given a calibrated camera angles. This help in calculating not only the location but also the angles of the objects in

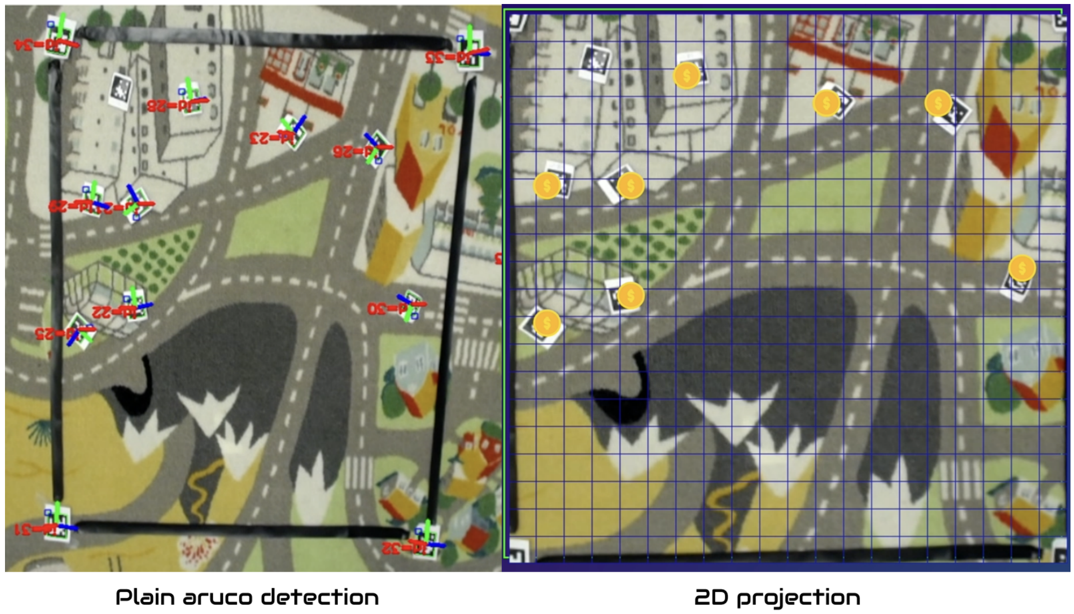


Figure 4-18: Aruco Projection. On the left, the plain Aruco marker detection, and on the right the 2D projection of the grid into the User Interface. You can see on the right that the corners of the projection are the centers of the markers designed as corners.

relation to each other. For this we had to compile OpencvJS from scratch, as the Aruco module was not available in the traditional links provided by OpenCV (e.g., https://docs.opencv.org/3.4.0/opencv.js).

After that, we just needed to analyze the camera stream, to identify all ArUco codes presented in the scene. Furthermore, to restrict the scene and the robots field of movement, we added 4 ArUco markers that represent the corners of the playing field (and therefore represent the virtual 2D grid), and then use those as a reference to match the location of any other ArUco code detected in our 2D grid. These corners helped us generate a projection matrix, that in turn can make us turn the input frame (which is usually tilted) to a 2D projection (that we use to show in the screen). An example of this can be seen in Fig 4-18

For the bots it was particularly challenging as we didn't only need information about where the Aruco marker was located, but in what direction it was moving. Technically this information should also be encoded in the Aruco detection, but we found it to be very flickery and simply unreliable. What we did instead was to have two Aruco Markers on top of each Doodlebot, and use the grid positions of both of them to deduce which direction the bot is looking at. Unfortunately, there were always some frames in which one of the markers on top of a bot was not detected, which made the angle detection also a bit inconsistent, but overall much better than what it was before.

Even though the Aruco detection was good at detecting markers, we found it to be very susceptible to different light conditions. If it was too dark or too bright, the system couldn't detect almost anything. For better detection, we found it useful to have the camera as low as possible (as long as the full grid is still in frame) and to have a soft light that is not too close to the grid or camera.

4.6.3 Applying move to physical bot

We already saw how we can call the method *apply_next_move_using_policies* to the virtual bots. However, to make the physical bots follow what the VirtualGrid dictates, a few considerations need to be taken.

Turning

For the most part, sending a command of "turn 90 degrees" works fine. However, sometimes the bots don't move all the way, due to signals getting lost, engines not working, or even just low battery. This can be problematic as one big assumption of our VirtualGrid is that bots only have 4 possible angles: 0 (looking right), 90 (looking up), 180 (looking left), and 270 (looking down). To mitigate this, before turning a 90 degrees angle, we check what's the current real angle, and if it's not within a certain threshold (we used 10 deg) of an axis, then we apply a turn first to correct it. However, as our bot angle detection wasn't the best, sometimes the "angle correcting" was way more than it should've. To keep correcting angles often enough so that our invariant is kept, but not too often so that the bot doesn't turn a lot, we decided to just do the angle correcting every certain amount of frames (we used 300).

Moving

"move 1" means two very different things if talking to the VirtualGrid than with the Physical Robot. They're in different units, so we needed to find a conversion to keep them consistent. From empyrical data, we found that

1. 28 units for the Physical Robot is equivalent to moving 5mm

2. The grid we had setup had a side length of 64mm, which consisted of 20 cells per side

With this information, we can deduce that our conversion from grid to physical coordinates can be calculated as $\frac{28units}{5mm} * \frac{64mm}{20cells} = \frac{18units}{cell}$, which means that for every "move 1" we receive from VirtualGrid, we should send a "move 18" to the robot to see the same effect.

4.6.4 Streaming Camera Frames

The purpose of the activity is to have two kids using two different computers. However, at most one laptop can be connected to a camera (which we need for the Aruco detection). To ensure that both students have the same camera feed in their interfaces, an IP Camera is needed. This is needed for receiving video frames remotely in real-time. For this purpose, we used open source code that claimed to be able to convert any camera into an IP camera.

We found the program extremely useful for our purposes, but very restrictive nonetheless. For example, if we want to see the images on two different remote computers we need to open up two different ports, as trying the same link twice breaks the stream, probably because the images were too heavy that the stream couldn't take two continuous receivers.

4.6.5 Summary

As we have collaboration be one of our main features, we spent a good amount trying what was the best way to assuring synchronization among different students' laptops, finding Firebase to be the most reliable and easier to scale. In this way, we made it possible for the students to be on the same page at all times, including our tutorial.

With a focus on having this interface be as simple and child-friendly as possible, we redesigned its initial iteration which had too many typing and clicking features, and use InteractJS instead to enable all the world-building experience to be through drag and dropping images into a grid, which we believe are much more intuitive, low-friction for kids.

For our purposes of teaching our curriculum, we are able to design an easy-to-understand set of options for Bot policies and movement types, that when changed affect the internal representation of the students' bots as well as the way they move. We hope this further exemplifies the importance of deciding on suitable policies and movement types for the bots.

Finally, with a focus on making the interface as engaging as possible, we designed different themes with the hopes of having as many students find a theme they really like.

Chapter 5

Results

5.1 Interface Results

One of the main research questions in this thesis is to investigate the interface design and its effectiveness in not only presenting the curriculum but also engaging the students. The interface was designed with several features including, 1) students' collaboration through synchronizing their actions, 2) simplicity and child-friendly, and 3) engaging interactions. We assess these aspects through interview/discussion questions, and survey questions, where we analyze the results as follows.

Interview Questions

In student interviews, we observed signs of excitement and engagement from students, with the most prominent aspects being able to create their own virtual world, and seeing the physical robot moves.

> "Mom, can we have this at home?" (S1)

We also saw how after the study they were able to make more connections about what they had learned and the real world.

> [When asked about their favorite part of the physical activity] "I like how it puts AI and the real world together, reminds me of Tesla!" (S2)

The design of the UI received mixed responses. While two students found the interface design a bit confusing, given all the options they were presented to them (policies, movement type), the rest of them found it more intuitive and easy to use.

Survey Questions

Overall students found the full activity engaging, regardless of which mode they did first. In 5-1 we can see that all the ratings for both modes are above 3/5 for every category, with 'Exciting' being the highest for both modes with a score of 4.5

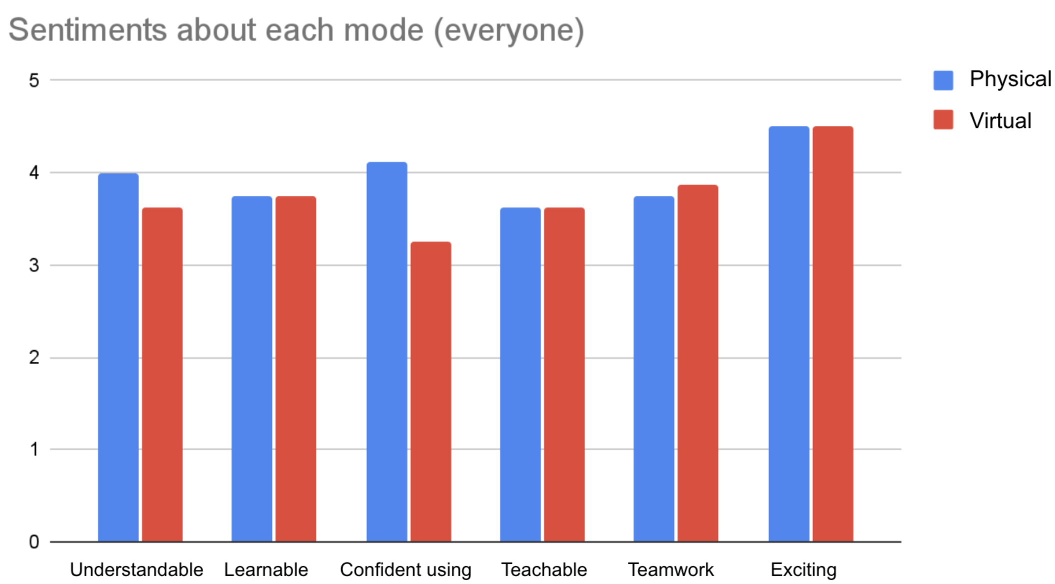


Figure 5-1: We asked students to give ratings from 1-5 to each mode given different metrics. Here are the average score each mode received from everyone.

We also explicitly asked them which mode (physical or virtual) they preferred across different categories, and even though they like both relatively similar, they have a slight preference for the Physical mode. More details can be seen in 5-2

Comparison of the order of final projects modes

We analyzed if there was any significant difference of the previous trends if we consider students from Groups A and B separately. In both cases, we found that kids had a tendency of preferring the last mode they played with. For example, people from

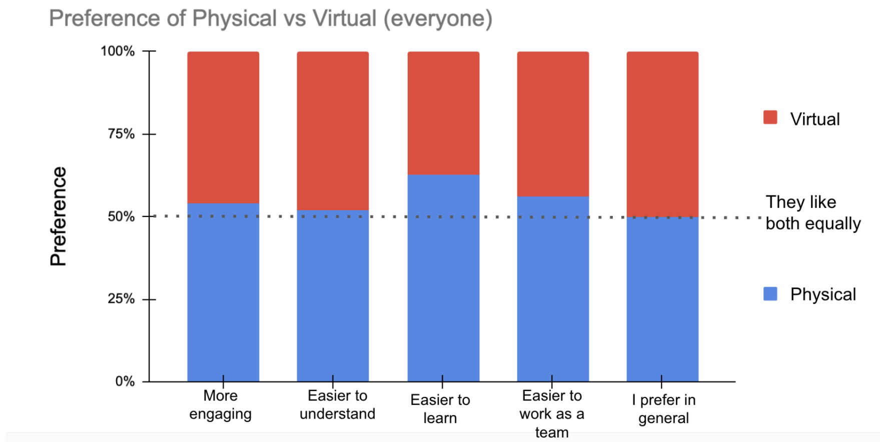


Figure 5-2: We asked students to choose which mode they preferred given different metrics. A higher score (presented to them as a 'Strongly agree' option) means they preferred the Physical mode better. Here are the average score each mode received from all the students.

group A (who started with the virtual mode), found the physical version better, and vice versa. These comparisons can be seen in figures 5-3 and 5-4

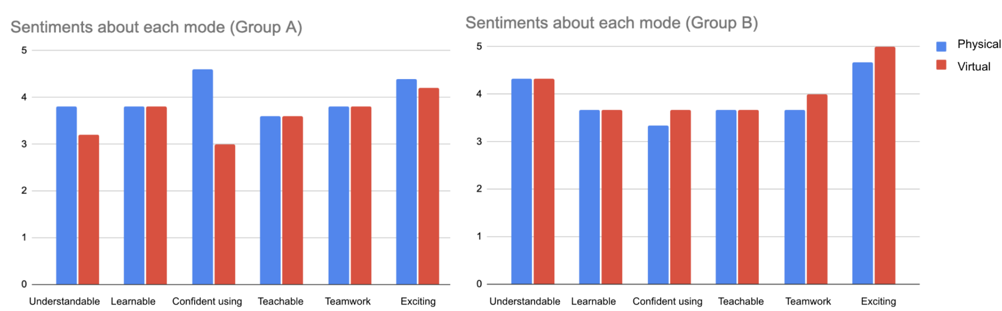


Figure 5-3: We asked students to give ratings from 1-5 to each mode given different metrics. Here are the average score each mode received from the ones in the Group A (Virtual first) and B (Physical first).

5.2 Education Evaluation

To evaluate the effectiveness of the proposed curriculum, we assess the students knowledge pre and post the interaction, as well as through expert observation. The results are presented in the following subsections.

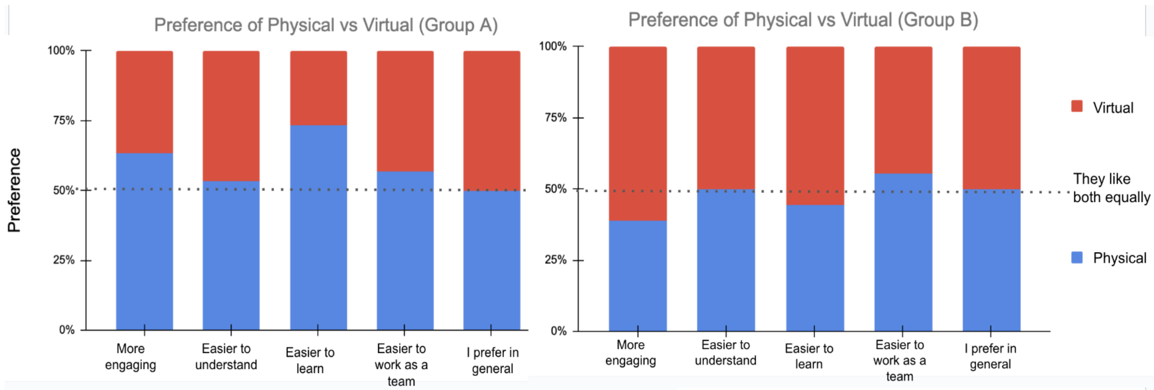


Figure 5-4: We asked students to choose which mode they preferred given different metrics. A higher score (presented to them as 'Strongly agree' option) means they preferred the Physical mode better. Here are the average score each mode received from the students in Group A (Virtual first) and B (Physical first).

Observations

We found the videos shown in the tutorial to be effective, as it was noted that three out of the 8 students received a 5/5 in the question "How would you rate the students' understanding of random vs. algorithmic path planning?", three received a 4/5 and the other two received 3/5. To support this conclusion, observers noted that some students responded to the 'Random video' question of "How long do you think it'll take for the random bots to reach their destination." with answers like "Veeery long!" or "too much!".

For the "How would you rate the students' understanding of creating maps?" question, five students scored a 5/5, and the other three got a 4/5. Observers mentioned that overall everyone seemed engaged and eager to create their maps in the virtual or physical mode. They also noted that students usually spent more time building their virtual environments whereas, for the physical part, they just spent a couple of minutes on it and then attempted to see the physical bots move.

Finally, for the "How would you rate the students' understanding of using AI for real-world societal impact" question, students got on average a 2/5. They mentioned that most students, when asked which theme to choose for their activity, usually chose the 'Pacman' theme (probably due to familiarity with it).

For the rest of the questions, there were not enough responses from the observers

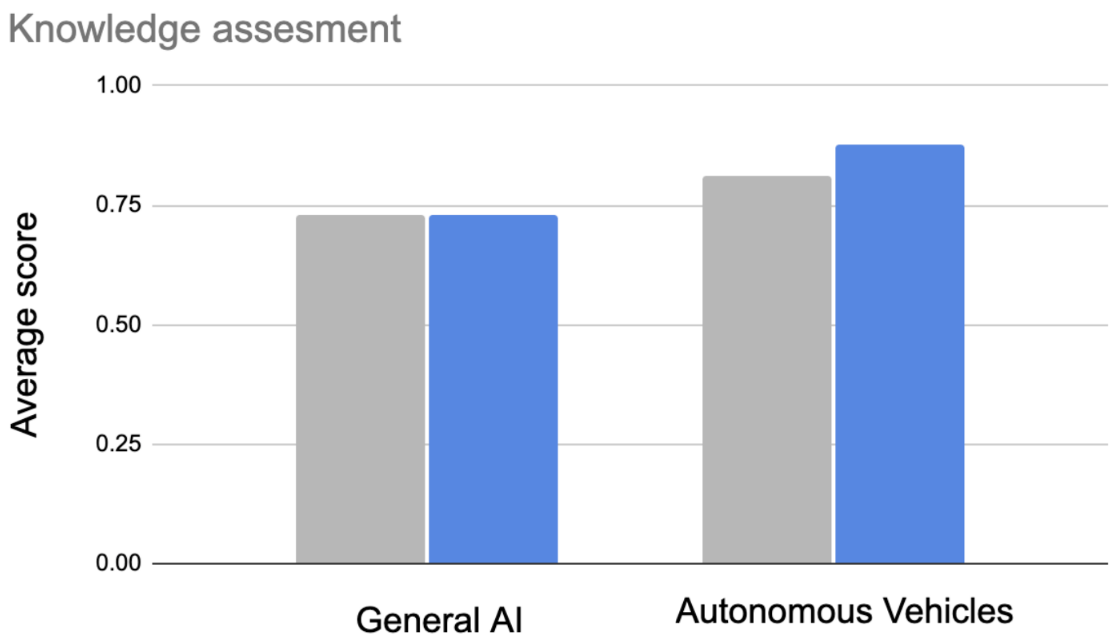


Figure 5-5: Comparison of knowledge about AI and Autonomous Vehicles concepts. The gray (left) bars represent before the activities, and the blue (right) bars represent after the full activity.

to give a useful summary.

Pre and Post questionnaire

To gather a sense of the level of knowledge students had before starting the activity, and to see how much that changed thanks to the activity, we had multiple choice questions about AI in general and about Autonomous Vehicles. In Figure 5-5 we can see that the average score of the students increased for Autonomous vehicles concepts, but not as much for AI in general.

On the other hand, not only in the technical side we cared about, but also on testing whether students feel more confident and connected more with the material taught. With that in mind, we created 5-5

We also found that the skill level of the kids increased, regardless of which group they were in. After the activity, kids felt more confident that they could apply what they learned in other scenarios, like designing a map that accurately reflects their world or planning the best path to reach a goal. These findings can be found in 5-7

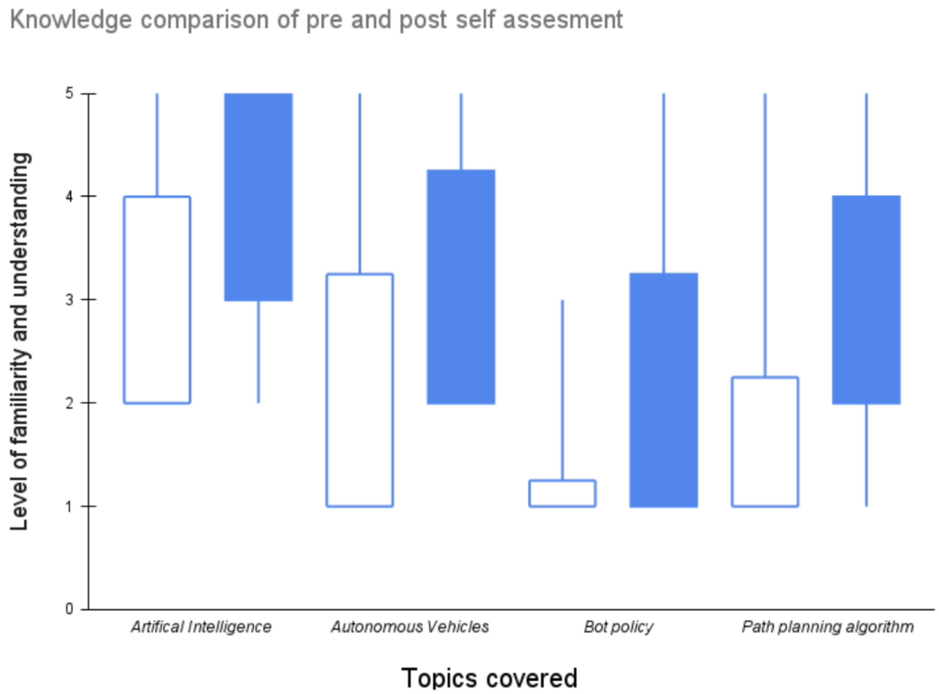


Figure 5-6: Comparison of self assesment about familiarity with different concepts. Hollow bars represent the pre-questionnaire, and the filled bars represent the post-questionnaire.

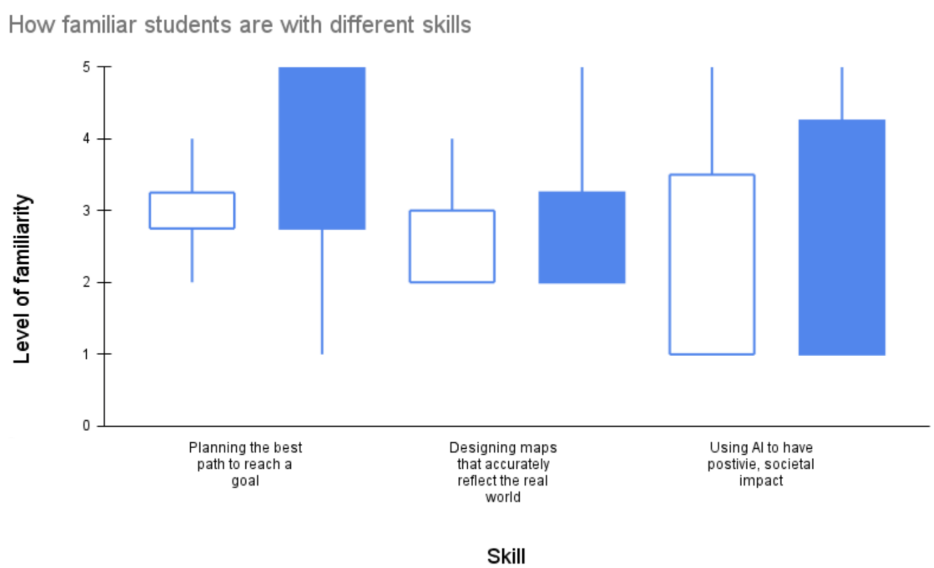


Figure 5-7: Comparison of self assessment about familiarity with different skills. Hollow bars represent the pre-questionnaire, and the filled bars represent the post-questionnaire.

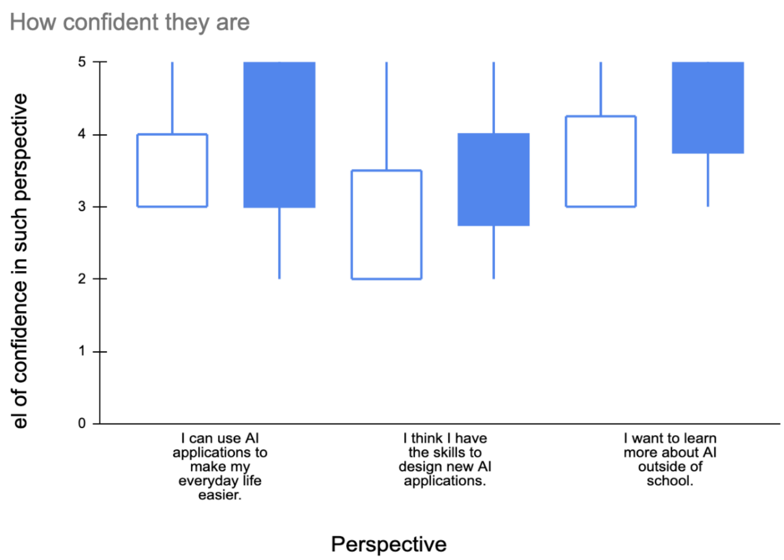


Figure 5-8: Comparison of self assesment about their perspective and confidence of using AI in their community. This shows significant improvement after the activity. Hollow bars represent the pre-questionnaire, and the filled bars represent the post-questionnaire.

Finally, their perspectives on AI and how they can use this on their day-to-day life increased, as can seen in 5-8

5.3 Summary

Student interviews reveal excitement and engagement and the UI design received mixed signals, with some finding it confusing but most liking the flexibility it provided.

Survey results show overall student engagement in both physical and virtual modes, with a slight preference for the last game played in their session. Students also felt more confident and skilled in applying what they have learned.

The curriculum's effectiveness is evaluated through pre and post-knowledge assessments, which showed an increase in knowledge in autonomous vehicles (what we covered), but not as much in AI in general.

Overall, the activities seem to have had a positive impact on kids's engagement, knowledge, and confidence about using AI.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

We presented an AI curriculum focused on autonomous navigation, with the use of a web-based User Interface as well as physical robots. An important focus of this study was to conclude whether an interface like this could be a useful and engaging way to teach AI. Through the use of student interviews, researcher observations, and pre/post-knowledge assessments, we noticed that not only did the student's knowledge about autonomous navigation increase but so did their confidence about using AI in their day-to-day.

The curriculum began with a tutorial, where kids watched short videos talking about Autonomous Navigation concepts followed by quick simulations of the User Interface that further explained the previous lesson. After the tutorial, the kids went through two types of activities: *virtual* where they were able to drag and drop objects using the UI; and *physical*, where they were able to move real bots and obstacles made of Legos. Both modes were made to give students the flexibility to build any playground they wanted. Some students experienced the *virtual* activity before the *physical*, and the rest the other way around.

We found that the order in which students did the activities influenced which one of them they ended up liking best. In particular, students that started with *virtual* tended to prefer the *physical* activity, and vice versa. We believe this to be partly because the last activity they did is usually fresher in their minds, and partly because of the unique experiences and advantages offered by each mode. This order effect suggests that both modes have distinct qualities that resonate differently with students, and exposing them to both modes can provide a well-rounded and enriching learning experience. Further investigation into the underlying factors contributing to these preferences could yield valuable insights for optimizing curriculum design and sequencing of activities.

In conclusion, the results indicate that the interface design and activities successfully engaged students, leading to improved knowledge and self-assessed skills. The tutorial effectively conveyed specific concepts related to autonomous navigation, while the activities facilitated practical application and confidence-building. Overall, this research contributes to the field of interface design for educational activities and provides valuable insights for designing effective and engaging learning experiences.

6.2 Future work

There are multiple ways this work could be extended. For example, while the activities proved effective in enhancing students' knowledge and confidence, some students found the interface design initially confusing due to the multitude of options presented to them. Simplifying the design or providing additional guidance could enhance usability and minimize potential barriers to engagement. On the other hand, increasing the library of themes and assets available to kids could improve their engagement as they'd have more flexibility in building what matters to them.

On the technical side, one challenge with using Aruco codes was that its detection depended heavily on the lighting conditions of the room. Using other types of detection like color detection or standard object detection could lead to better results.

Finally, the published website currently works for the *virtual* mode and the *physical* mode assuming the kids are in the room with the bots. It'd be a great improvement to allow the *physical* mode to be played by students remotely, to increase the number of students that could be benefited by this program.

Chapter 7

Resources

The survey questions were designed by Randi Williams. Website image assets were designed by Garrett Beazley. The thesis composition was written on Overleaf with proofreading help from ChatGPT.

Appendix A

Pre and Post questionnaire

Autonomous Vehicle Questionnaire

Sign in to Google to save your progress. Learn more

Pre-Questionnaire

We would like to start off by knowing how much you already know about artificial intelligence and how interested you are in the topic. If you don't know how to answer any of the questions, don't worry. Do your best. We will not share your answers with others.

How familiar are you with the following keywords?

	1 - I've never heard of this	2 - I’ve heard of this but only understand it a little	3 - I am beginning to understand this, but still need some help	4 - I understand this well and can talk about it without help	5 - I understand this very well and can teach it to someone else
Artificial Intelligence	○	○	○	○	○
Autonomous vehicles	○	○	○	○	○
Bot policy	○	○	○	○	○
Path planning algorithm	○	○	○	○	○

Which of the following statements is true about AI?

- ○ Any robots that look like and talk like humans are considered Artificial Intelligence.
- ○ AI is another word for systems that can verbally interact with humans, like Alexa or Siri.
- ○ AI is the simulation of human intelligence by computers and machines.
- ○ All robots created by humans are considered Artificial Intelligence.

Look at the following technologies and indicate whether or not they involve AI.

	AI	Not AI	I am unsure
A machine that washes clothes.	○	○	○
A system that adds large numbers.	○	○	○
An app that finds the shortest path from your current location to your destination.	○	○	○
A feature in an email client that recognizes whether a message is spam (junk mail) or not.	○	○	○
A car that you can drive around using a remote control.	○	○	○

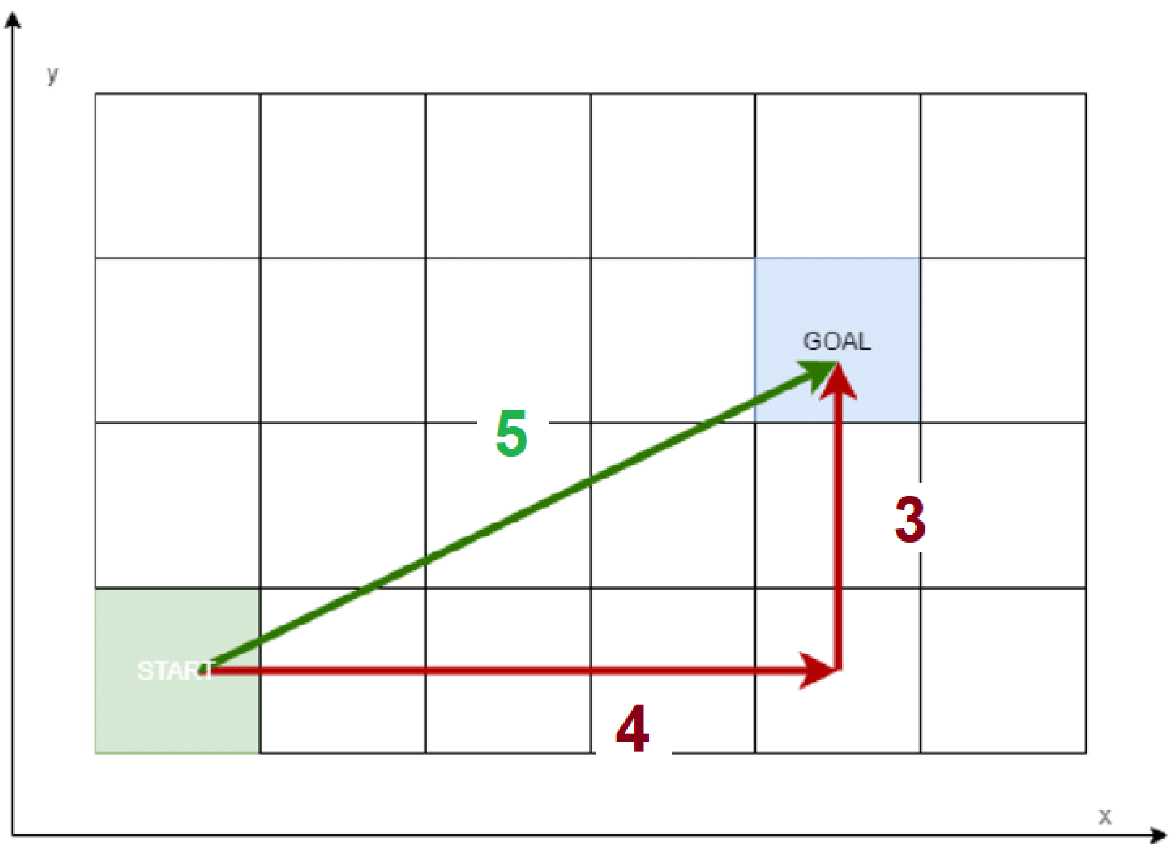
How familiar are you with the following skills?

	1 - I've never heard of this	2 - I've heard of this, but am not sure I can do it	3 - I am beginning to understand this, but need help to do it	4 - I understand this well and can do it on my own	5 - I understand this very well and show someone else how to do it
Using AI to have postivie, societal impact	○	○	○	○	○
Designing maps that accurately reflect the real world	○	○	○	○	○
Planning the best path to reach a goal	○	○	○	○	○

Which of the following statements are true about autonomous vehicles?

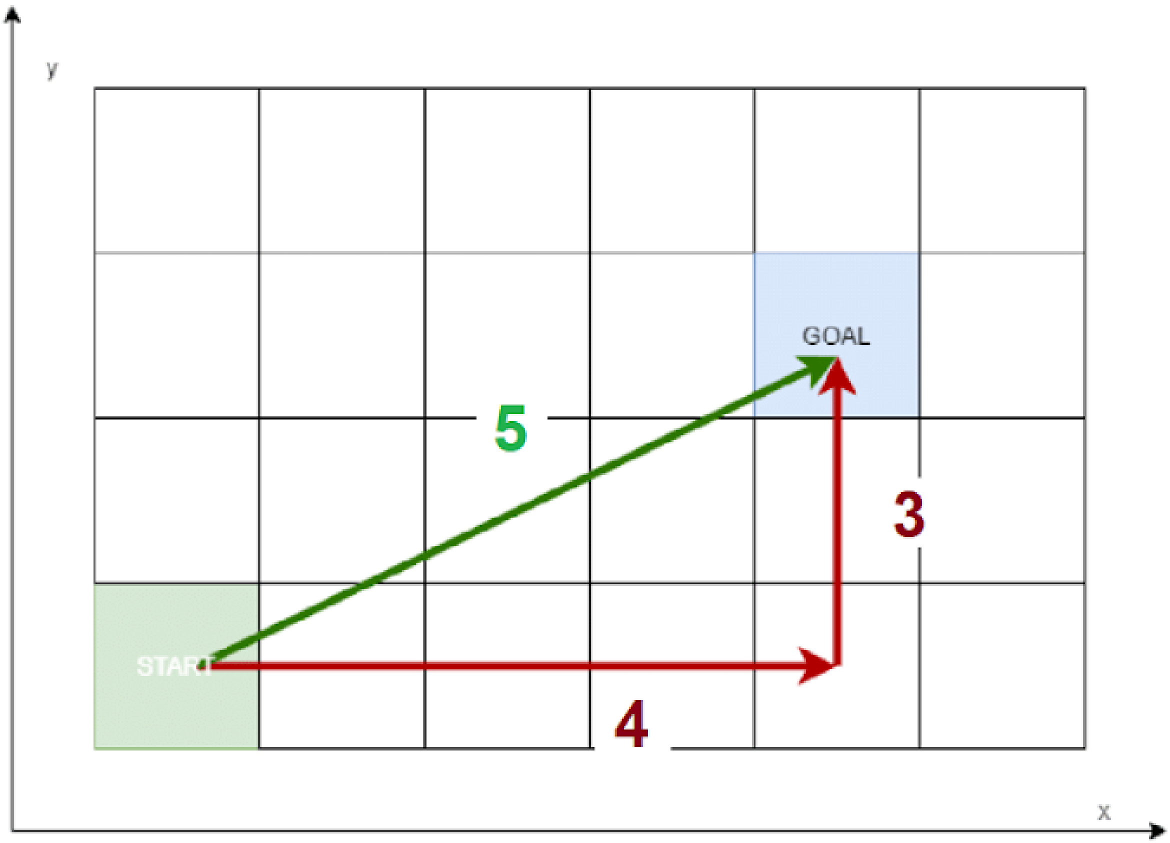
- ○ Autonomous vehicles use sensors to safely avoid obstacles and plan paths to their goals.
- ○ There are no fully autonomous vehicles currently being used today.
- ○ With AI, autonomous vehicles learn to drive by studying humans and exactly copying how they drive.
- ○ Using a random path planning policy for autonomous vehicles is often faster than other policies.

Manhattan distance, also called city block distance, is the distance between two points along a grid where you can only drive along a grid. In the following image, what is the Manhattan distance from the start to the goal?



- The Manhattan distance from Start to Goal is 12.
- The Manhattan distance from Start to Goal is 4.
- The Manhattan distance from Start to Goal is 7.
- The Manhattan distance from Start to Goal is 5.

Euclidean distance is the straight line distance between two points. In the following image, what is the Euclidean distance from the start to the goal?



- The Euclidean distance from Start to Goal is 4.
- The Euclidean distance from Start to Goal is 3.
- The Euclidean distance from Start to Goal is 7.
- The Euclidean distance from Start to Goal is 5.

Which of the following are realistic benefits and harms of autonomous vehicles.

- [ ] Harm: Autonomous vehicles will create too many jobs in the economy, and there won't be enough people to fill them.
- [ ] Benefit: Autonomous vehicles can be used in dangerous situations (ex: war, space, construction), instead of humans.
- [ ] Benefit: Autonomous vehicles are perfectly safe and secure. They will be able to work without humans checking them.
- [ ] Harm: A vehicle that is smart enough to drive itself is likely smart enough to plan a revolt against humans.
- [ ] Harm: Autonomous vehicles could unpredictably make mistakes when unexpected things happen.

How much do you agree with the following statements?

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I am curious about new uses of AI in our society.	○	○	○	○	○
I will talk to members of my household about what I know about AI.	○	○	○	○	○
I'd like to learn AI so I can help other people.	○	○	○	○	○
I do not think I have the skills to design new AI applications.	○	○	○	○	○
I would take a class about AI if it is offered in my school.	○	○	○	○	○
I want to learn more about AI outside of school.	○	○	○	○	○
I can imagine possible future uses of AI.	○	○	○	○	○
I can use AI applications to make my everyday life easier.	○	○	○	○	○

Next

Clear form

Appendix B

Project feedback form

How much do you agree with the following statements?

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I was unsure how to use the physical Doodlebot system.	○	○	○	○	○
It was easy to work as a team with the physical Doodlebot system.	○	○	○	○	○
I feel confident showing someone else how to use the physical Doodlebot system.	○	○	○	○	○
The physical Doodlebot system was difficult to understand.	○	○	○	○	○
Using the physical					

physical Doodlebot system was exciting and engaging.	○	○	○	○	○
Someone can learn how to use the physical Doodlebot system quickly.	○	○	○	○	○

system quickly.

After trying both the physical and virtual Doodlebots, how much do you agree with the following statements?

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The virtual Doodlebots were more exciting and engaging than the physical Doodlebots.	○	○	○	○	○
The physical Doodelbots were more difficult to understand than the virtual Doodlebots.	○	○	○	○	○
Someone could learn how to use the physical Doodlebot system more quickly and easily than the virtual Doodlebots.	○	○	○	○	○

It was easier to work as a team with the physical robots than the virtual robots.	○	○	○	○	○
I prefer the virtual Doodlebots to the physical Doodlebots.	○	○	○	○	○

Do you have any additional thoughts about how the physical Doodlebot system was compared to the virtual Doodlebot website?

Your answer

Back Next Clear form

Appendix C

Observation form

Researcher Evaluation Form
Use this form to evaluate the participants as they go through the study. Record their codenames here.

Participant 1:	
Participant 2:	

Activity Observation. As the participants complete the activities and projects, make note of examples of their learning or misconceptions they seem to have. At the end of the activities and projects, give their understanding an overall rating.

Rating *(fill out after the activities and projects are done)*	**Positive Examples of understanding** *(fill out as participants complete the activities and projects)*	**Negative Examples of misunderstanding** *(fill out as participants complete the activities and projects)*
1 - They make a lot of mistakes and/or need a lot of help to discuss and use these ideas. 5 - They use and explain the ideas correctly and thoroughly with minimal support.		
How would you rate the students' understanding of Artificial Intelligence? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		
How would you rate the students' understanding of autonomous vehicles? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		
How would you rate the students' understanding of Euclidean and Manhattan distance? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		

Rating *(fill out after the activities and projects are done)*	**Positive Examples of understanding** *(fill out as participants complete the activities and projects)*	**Negative Examples of misunderstanding** *(fill out as participants complete the activities and projects)*
How would you rate the students' understanding of random vs. algorithmic path planning? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		
How would you rate the students' understanding of different bot policies? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		
How would you rate the students' understanding of creating maps? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		
How would you rate the students' understanding of using AI for real world societal impact? Participant 1 1 2 3 4 5 Participant 2 1 2 3 4 5		

Appendix D

Interview questions

Project 1 Interview or Discussion. Ask students about their project. Try to record exact wording and note who is giving each response.

1. What does your project do?	
2. What inspired your project?	
Here is a list of the topics and skills we explored today: **Artificial Intelligence**, **autonomous navigation**, **Euclidean and Manhattan distance**, **bot policy**, **Planning the best path**, **Designing maps**, **Using AI for positive societal impact**,	
3. Which topics show up in your project and how?	
4. What was your favorite part of doing this project?	
5. How would you extend your project if you had more time?	

Project 2 Interview or Discussion. Ask students about their project. Try to record exact wording and note who is giving each response.

1. What does your project do?	
2. What inspired your project?	
Here is a list of the topics and skills we explored today: **Artificial Intelligence**, **autonomous navigation**, **Euclidean and Manhattan distance**, **bot policy**, **Planning the best path**, **Designing maps**, **Using AI for positive societal impact**,	
3. Which topics show up in your project and how?	
4. What was your favorite part of doing this project?	
5. How would you extend your project if you had more time?	

Activity Feedback. Ask students their thoughts about the physical and virtual robots. Try to record exact wording and note who is giving each response.

Website / System Feedback Participants used the (circle one): physical Doodlebot system or virtual Doodlebot website	
1. Did you understand more with the physical Doodlebot system or the virtual Doodlebot website?	
2. Did you have more fun with the physical Doodlebot system or the virtual Doodlebot website?	
3. What, if anything, would you change about either the physical or virtual Doodlebots to make it better?	

Bibliography

[1] Theofylaktos Anastasiadis, Georgios Lampropoulos, and Kerstin Siakas. Digital game-based learning and serious games in education. *International Journal of Advances in Scientific Research and Engineering (IJASRE), ISSN:2454-8006, DOI: 10.31695/IJASRE*, 4(12):139–144, Dec. 2018.

[2] A. Bezgodov, A. Karsakov, K. Mukhina, D. Egorov, and A. Zakharchuk. Learning ai techniques through bot programming for a turn-based strategy game. In *Proceedings of the European Conference on Games-based Learning*, page 66–74.

[3] Noelene Callaghan. Investigating the role of minecraft in educational learning environments. *Educational Media International*, 53(4):244–260, 2016.

[4] Thomas K. F. Chiu, Helen Meng, Ching-Sing Chai, Irwin King, Savio Wong, and Yeung Yam. Creation and evaluation of a pretertiary artificial intelligence (ai) curriculum. *IEEE Transactions on Education*, 65(1):30–39, 2022.

[5] Jennifer de Castell, Suzanne ; Jenson. Digital games for education: When meanings play. *Intermédialités / Intermediality*, (9):113–132, 2007.

[6] A. Elliot. *Digital technologies and social interaction.* Routledge, 2018.

[7] Gerald Futschek and Julia Moschitz. Learning algorithmic thinking with tangible objects eases transition to computer programming. In Ivan Kalaš and Roland T. Mittermeir, editors, *Informatics in Schools. Contributing to 21st Century Education*, pages 155–164, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[8] Michail Giannakos, Iro Voulgari, Sofia Papavlasopoulou, Zacharoula Papamitsiou, and Georgios Yannakakis. *Games for Artificial Intelligence and Machine Learning Education: Review and Perspectives*, pages 117–133. Springer Singapore, Singapore, 2020.

[9] Philip Hingston, Barbara Combes, and Martin Masek. Teaching an undergraduate ai course with games and simulation. In *Edutainment*, 2006.

[10] Xiaodong Huang. Aims for cultivating students'key competencies based on artificial intelligence education in china. *Education and Information Technologies*, 26(5):5127–5147, 2021.

[11] Seonghun Kim, Yeonju Jang, Woojin Kim, Seongyune Choi, Heeseok Jung, Soohwan Kim, and Hyeoncheol Kim. Why and what to teach: Ai curriculum for elementary school. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):15569–15576, May 2021.

[12] Liang-Yi Li, Chih-Wei Chang, and Gwo-Dong Chen. Researches on using robots in education. In Maiga Chang, Rita Kuo, Kinshuk, Gwo-Dong Chen, and Michitaka Hirose, editors, *Learning by Playing. Game-based Education System Design and Development*, pages 479–482, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[13] Minhua Ma, Andreas Oikonomou, and Lakhmi C. Jain. *Innovations in Serious Games for Future Learning*, pages 3–7. Springer London, London, 2011.

[14] Nitin Naik. A comparative evaluation of game-based learning: Digital or non-digital games?, 10 2014. Copyright - Copyright Academic Conferences International Limited Oct 2014; Document feature - ; Tables; Last updated - 2015-04-26.

[15] Nitin Naik. Non-digital game-based learning in the teaching of mathematics in higher education, 10 2014. Copyright - Copyright Academic Conferences International Limited Oct 2014; Document feature - Tables; ; Last updated - 2015-04-26.

[16] T. Nantsou, G. Frache, E.C. Kapotis, H.E. Nistazakis, and G.S. Tombras. Learning-by-doing as an educational method of conducting experiments in electronic physics. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 236–241, 2020.

[17] Alpay Sabuncuoglu. Designing one year curriculum to teach artificial intelligence for middle school. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '20, page 96–102, New York, NY, USA, 2020. Association for Computing Machinery.

[18] Bawornsak Sakulkueakulsuk, Siyada Witoon, Potiwat Ngarmkajornwiwat, Pornpen Pataranutaporn, Werasak Surareungchai, Pat Pataranutaporn, and Pakpoom Subsoontorn. Kids making ai: Integrating machine learning, gamification, and social context in stem education. In *2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pages 1005–1010, 2018.

[19] Josh Trout and Brett Christie. Interactive video games in physical education. *Journal of Physical Education, Recreation & Dance*, 78(5):29–45, 2007.

[20] Alf Inge Wang and Rabail Tahir. The effect of using kahoot! for learning – a literature review. *Computers & Education*, 149:103818, 2020.

[21] Randi Williams, Hae Won Park, Lauren Oh, and Cynthia Breazeal. Popbots: Designing an artificial intelligence curriculum for early childhood education. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):9729–9736, Jul. 2019.