ARBlockbot: Accessible Robotics and Programming Education in AR

Kevin Feng*[1], Daniel Martin*[2], Anne Liu[3], Anjali Thatte[4]
[1]Princeton University, [2]Georgia Tech, [3]Queen's University, [4]University of Waterloo
Supervised by Prof. Anand Bhojan

Motivation

Robotics education often requires heavy reliance on hardware. Some schools that do not have resources to purchase, maintain, and store the necessary hardware may difficulty exposing students to robotics. Even if the school has required hardware, students who live further away from school may be disadvantaged.

Hardware setup and debugging is often strenuous, which may not be ideal when prototyping and testing rapidly. Additionally, if special circumstances such as a pandemic prevents in-person collaboration, traditional robotics education may not even be possible.

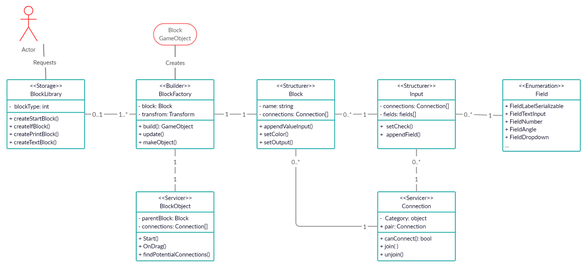
We present ARBlockbot, an educational robotics and programming app for Android and iOS phones + tablets, geared towards primary and middle school students. ARBlockbot eliminates the need for a physical robot or any hardware equipment and allows the virtual robot to interact with real-world objects and surfaces without AR markers.

Previous Work

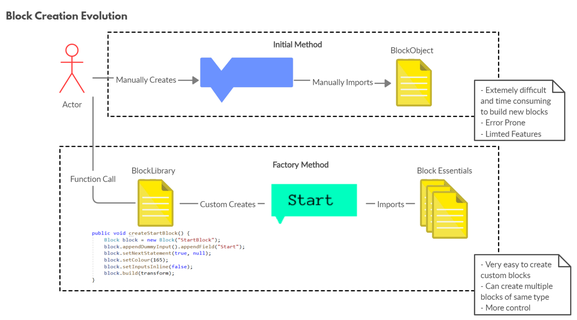
Some existing commercial platforms for robotics education, such as [1], do not require the use of physical hardware, but are based in an entirely artificial world. [2] demonstrates that AR can be successfully deployed to assist middle school students learn basic robotics. [3] uses and block-based programming in an AR-based robotics learning system, but requires the scanning of physical marker cards in order to activate the code editing interface and many other functions. [4] also combines block programming and an AR-based robot, but robot movement is restricted to a physical table with assistive markers.

Methodology

Using graphic language from Google's Blockly block programming library, we rebuilt the main Blockly functionalities in C# and Unity. We engineered a pipeline for blocks to be created upon the user's request and link blocks to code that would turn the wheels of a 3D model of an RC buggy. An overview of the architecture of our system is shown below:



We then iterated on our system to make it more flexible and scalable. We built a Block-Factory to create blocks from a BlockLibrary and import graphical components from Block-Essentials. The evolution of our system is shown below:



We designed and built our user interface in Unity. We used RC vehicle prefabs from SteamVR, which allowed us to control each wheel separately. We used ARFoundation's default packages for plane detection.

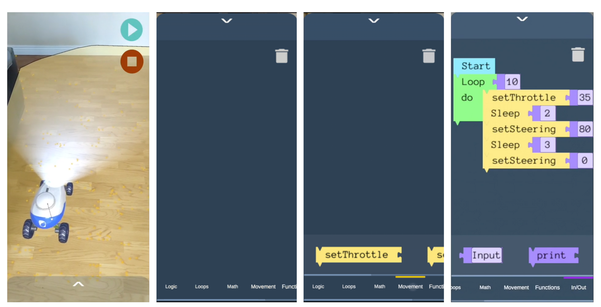
Results

The virtual robot, which is placed wherever a user taps on a detected plane, is able to move forward and back, turn any angle, and obey conditionals and loops as specified by the code blocks. The robot follows the laws of physics: it can hit an object and roll over, fall down from one surface onto another, and skid while turning or braking.

We found that our initial UI, which was based on that of Blockly's, was not well-optimized for phones and tablets. We redesigned the UI to be minimal, hiding most controls whenever possible and allowing for maximum camera view. To display diverse block selections with clarity on a small screen, we employed layers of collapsible horizontally scrolling menus. We performed a very small, COVID19-safe user testing session with mostly family and relatives. The new UI was heavily favoured:

Question	UI	User 1	User 2	User 3	User 4	User 5
Successfully identified purpose of UI and/or app	Old UI	x	✓	x	x	x
	New UI	✓	✓	✓	✓	✓
Were able to navigate the UI to build the "go forward" command	Old UI	x	N/A	x	x	N/A
	New UI	✓	N/A	✓	x	N/A
Prefers which UI	Old UI					
	New UI	✓	✓	✓	✓	✓

Screenshots from the app, complete with the new UI, are below:



Conclusion & Future Work

We are glad that the motion of the virtual robot can be precisely controlled using the generated code blocks. Our improvement in user interface was significant, and allowed more information to be displayed in a more accessible way with more clarity and less obtrusiveness.

Our immediate next steps would be to refine the look and presentation of the code blocks. We also would like to incorporate multiple types of robots besides a simple RC vehicle, such as a drone that can fly or a humanoid robot that can walk and jump. For a more enhanced classroom experience, a multiplayer ode can be added so students can join AR "rooms" to complete tasks or hold competitions with their robots.

References

[1] VexcodeVR. https://vr.vex.com/.
[2] Cheli, M., et al. (2018). "Towards an Augmented Reality Framework for K-12 Robotics Education." *VAM-HRI* 2018.
[3] F. Ou Yang, "The Design of AR-based Virtual Educational Robotics Learning System," *IIAI-AAI* 2019.
[4] G. Stein and A. Lédeczi, "Mixed Reality Robotics for STEM Education," *B&B* 2019.

Acknowledgements

We would like to thank Prof. Bhojan and our TA Liu Rui for their support and feedback throughout this semester, especially given the current pandemic circumstances. We would also like to thank our user testers for their time and comments.