LET'S TRAIN VIRTUAL ROBOTS

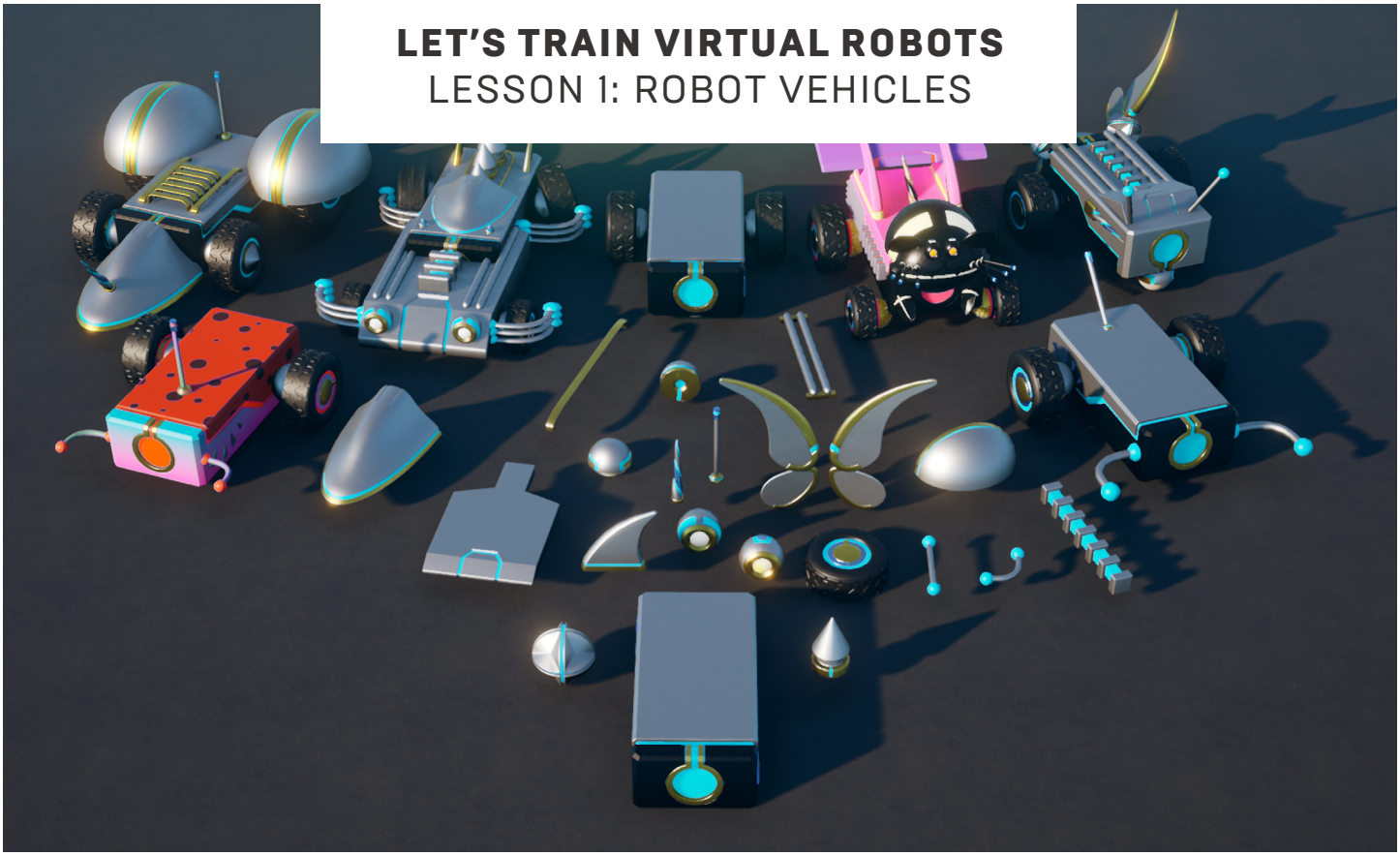
LESSON 1: ROBOT VEHICLES

UNREAL ENGINE

LESSON PLAN

TABLE OF CONTENTS

LET'S TRAIN VIRTUAL ROBOTS
LESSON 1: ROBOT VEHICLES



LESSON CLASS GUIDE INFORMATION

Lesson Title: Let's Train Virtual Robots: **Robot Vehicles**
Subject: Computer Science, Engineering
Grade: 8–12 (students must be 13 or older to participate in this class)
Lesson Timeframe: One hour
Student Guide

SELECTING ACTIVITIES

These robotics learning activities can be facilitated in any sequence, but it is recommended to proceed sequentially from activity 1 through 5. The lessons work together, building on prior skills, so that a student who completes all five robotics learning activities will learn the basics of coding autonomous robots and how to use the Learning Kit for future activities. It is recommended that all students start with the first activity in order to learn the features of the Learning Kit and how it supports the creation of a robot as well as the coding of motors and sensors in the Unreal Engine environment.

Let's Train Virtual Robots explores the concepts of robotic engineering using applied physics. While giving students immediate feedback on their coding and providing principles/coding language knowledge that they can apply to other physical robotic systems they may work with in the future.

TECHNOLOGY NEEDS

This Unreal Learning Kit requires the installation of the Unreal Engine software and the Unreal Learning Kit files. The minimum hardware requirements are :

- Windows 10 64-bit computer
- Quad-core Intel or AMD, 2.5 GHz or faster processor
- DirectX 11 or DirectX 12 compatible graphics card
- 256 GB SSD (OS Drive)
- 2 TB SSD (Data Drive) <TO DEVELOP GAMES ON UE>
- 8 GB RAM

You will also need an administrator or owner role to install the software.

Installation and setup should be completed prior to beginning the first lesson. For help with installing Unreal Engine, click here.

For help with technical requirements, see this site.

This is a virtual robotic system; no additional robotic parts or kits are required.

AUTHOR CONTACTS

Brian Dickman, Cleverlike Studios

- Author: Brian Dickman
- Email: brian@cleverlike.com
- Twitter: @cleverlike
- LinkedIn: https://www.linkedin.com/in/cleverlike/

Rebecca Parrent, Cleverlike Studios

- Author: Rebecca Parrent
- Email: rebecca@cleverlike.com
- LinkedIn: https://www.linkedin.com/in/rebecca-parrent-5214168/

Ian Southwell, Cleverlike Studios

- Author: Ian Southwell
- Email: Ian@cleverlike.com
- Twitter: @iansouthwell2
- LinkedIn: https://www.linkedin.com/in/southwellian/

Author Brian Dickman studied computer science and operates a full-time game development studio that produces entertaining and educational content inside of popular video games.

Author Rebecca Parent has a Bachelor of Science in Music and Media, along with a Master of Education in Curriculum Design with a STEM focus. Rebecca's work experience includes training adults in computer use and database custom development, and training youth in technology applications and coding.

Rebecca and Brian teach robotics construction and coding as an engaging fun activity that connects to real-world coding languages, coding protocols, and best practices in coding and engineering design. Both authors have shepherded several student teams through robot competitions, specifically in the FIRST LEGO League, FIRST Tech Challenge, and VEX competitions.

Author Ian Southwell attended the Academy of Art University in San Francisco before deciding to take his love for 3D virtual worlds and make it a career. After graduating with a Bachelor in Fine Arts for Computer Animation, he has worked for over a decade as a designer, instructor, and artist. During this time, he worked with seven different studios, gaining an advanced working knowledge of building a game from first concept to final bug testing. He also understands the pipeline of product design, from initial design sketches through a digital sculpt in ZBrush to a 3D-printed object to a cast metal product. He continues to take classes to expand his knowledge and skillsets.

LEARNING KIT WITH UNREAL ENGINE ROBOTICS

Physical robotic systems can be expensive to purchase and difficult to maintain in a busy learning environment. This virtual robotic kit will give students access to a robot that performs with real applied laws of physics, looks like common instructional robot designs and develops skills that can be applied to many robot systems regardless of future hardware and coding languages used.

Robotics is an excellent platform for learning to code because students can see their coding in action. An obedient robot that performs exactly as instructed will provide learners with humorous mishaps and well-earned victories. These activities along with the Unreal Learning Kit will provide a uniquely creative and entertaining coding experience for all students. The simulated physics in Unreal Engine helps students experience the nuances of real-world robotics programming.

Facilitating the Activity

In preparation for facilitating each activity, a video walk-through of the activity is provided by the activities' creators. These walk-through videos are helpful in showing how the lesson can be facilitated and may also be a resource for the students to follow independently.

Through most of the activities, a new concept will be introduced, and students will be able to run a live demonstration of the key learning elements of this concept. In many cases, the demonstration will be followed by a student activity to reinforce the learning by doing the work. The facilitator should review the concept and get student observations/connections/reflections on each concept before moving to the next topic.

The process of introducing a concept, experiencing a live demonstration, and applying the learning to their project will be repeated throughout each robotics learning activity. Upon completion of the activities, extension activities will be suggested for advanced students or groups that want to extend the activity beyond an hour.

The facilitator should be available for guiding problem-solving as needed by asking questions. What were you trying to accomplish? What steps did you take? What did you observe? What can you try next? Review the Engineering Process for inspiration.

Students can work independently, or in small groups as needed. Consider recruiting the assistance of students that have shown proficiency with the activities to help aid their peers.

ACTIVITY ONE – ROBOT MOVEMENT

Introduction

When did you last clean your room? Do you wish you had a robotic vacuum to pick up those old snacks from last week? And should it avoid vacuuming up your cat and your earbuds?

Perhaps we could even have an outdoor robotic vacuum to clean your community. Today's robotic vacuums have many complex commands simplified to automated movements and actions to clean quickly and effectively every time. How did the engineers and programmers determine the movements needed? How did they code these movements?

Activity Overview

In this activity, you will learn how to code a two-wheeled robot to use tank movement for navigating to specific targets with a minimal number of commands. The virtual robot in this activity will perform like a robotic vacuum used in your home, in that it can move forward, backward, turn right, turn left, and spin around, all while being designed to not tip over. We will also explore the engineering and physics principles of a well-designed robot, regarding its weight, the location of its wheels, and a gliding ball for a stable robot that can produce consistent results.

In order to program a robot to accomplish a task, one must understand and communicate output actions. This communication must be accomplished using a specific software language that both the student and the robotic device will understand. Students will understand how to use the Blueprint node-based coding in the Unreal Learning Kit to control their robot's movement.

Students will construct program commands, test their robot, and then return to their coding to adjust any instruction setting that did not accomplish their given goal. They will execute their code on the virtual robot until the goal is accomplished.

Coding Connection

This first activity will get students familiar with navigating and building content inside of Unreal Engine. This is the game development platform used to create popular video games like *Fortnite*, *Rocket League*, and many more.

Essential Questions

1. How does this robot's design impact its' driving and turning?
    a. How does the location of the wheels affect its balance?
    b. How does wheel placement in the robot design affect its performance?
2. How does the initial placement/position of the robot impact its outcome?
3. What settings must be set to engage the motors to drive the robot?
4. What methods can be used to turn a two-wheeled robot?
5. What limitations exist when coding an autonomous robot without any input from sensors or humans? (Dead Reckoning – navigating without real-time input)

Learning Outcomes

1. Understand the basic navigation of the Unreal Engine platform, tools, and dashboard
2. Open a project in Unreal Engine
3. Edit and duplicate a blueprint in Unreal Engine
4. Edit the motor settings to drive and turn the robot
5. Understand the design benefits of a two-wheeled robot vehicle with a third touchpoint
6. Predict and calculate the code and settings needed, based on known robot design factors, prior to running the program

Getting Started in Unreal Engine

In order to be successful with this Unreal Learning Kit activity, participants will need to learn the basics of navigating the Unreal Engine interface. Improved proficiency in navigating the player/camera and locating the panels/windows inside the application will create a richer learning experience. Each activity will review the essential interface components and controls needed for completing the exercise.

If your participants are not familiar with Unreal Engine, consider allocating more time for the initial activity or plan some preparation exercises prior to starting the activity.

Scope of Unreal Engine Usage

- Launching Unreal Engine and opening the sample project
- Identifying and activating various levels/maps
- Playing your code and exiting Play mode
- Navigating your camera/view in the Viewport
- Creating your own robot by duplicating the original
- Moving and rotating actors in a level
- Editing default robot settings
- Saving your project

LEARNING CONCEPTS

Robot Construction

Since a robot is typically a physical object operating in an environment, the way it is designed will impact its operation. We will start by reviewing the essential components of a well-designed robot. We are focusing all activities on a two-wheeled robot that has a low-friction gliding ball as its third point of contact to the ground.

Balance

Review how wheel placement can impact the balance of the vehicle, making it easier to fall over if done incorrectly.

Demonstration: See demonstrations of this in the sample project.
(Optional: Add a discussion about center of gravity if applicable)

Traction

Review how traction is impacted by the vehicle design. Locating wheels near the mass of the vehicle body will increase friction, helping the wheels make better contact with the ground.

Demonstration: See demonstrations of this in the sample project.

ACTIVITIES

Robot Movement

A robot with two independent motors must be controlled by sending speed and direction values to each motor. This section will explore how to move the robot forward, backward and make various types of turns.

Controlling the Motors

- Identify the name and location of each motor relative to the vehicle's front and back
- Set a value to define the speed and direction of movement for each motor
- Determine how the motor directions relate to the movement direction of the vehicle
- Adjust the amount of time the motor is active to control the distance traveled

Forward and Backward Movement

- **Activity:** Edit the robot's code to make it move forward and backward

Turning with Tank Steering

- **Demonstration:** Observe various types of turns in action. (Spin Turn, Pivot Turn, Arc Turn)
- **Activity:** Code your robot to do a pivot turn
- **Activity:** Code your robot to do a spin turn
- **Activity:** Code your robot to do an arc turn
- **Demonstration:** Observe the lack of consistency when navigating a robot without any feedback from sensors or interaction from the student. This exercise is a setup for activity 2, which introduces different motor controls based on real-time sensor feedback

CHALLENGE ACTIVITY

Now that the student has learned how to code the robot to move and turn, it's time to put that learning into practice. The challenge level will present multiple targets. The student must code the robot to reach a target using a single arc turn. The student will set the motor speed, direction, and duration before running each test. Engineering process should be encouraged: predict, test, observe, and iterate. Students should log attempts and use the data to make informed decisions for each subsequent attempt. This process will get the students comfortable with navigating the Unreal Engine environment and manipulating the code to control their robot.

Extension Activities / Advanced Challenges

- Create more complex paths by stringing together multiple commands
- Try reaching the target in the fastest possible time and observe the new challenges that arise when increasing speed
- Code your robot to perform a victory dance!
- Consider what solutions could improve the accuracy and reliability of your robot instructions
- Discuss variables that impact performance (wheel size, surface friction)
    - Discuss two-wheel models versus four-wheel models

Activity Review

- Open and navigate Robotics Learning Kit in Unreal Engine
- Balance robot design
- Improve traction in robot design
- Coding the robot to move forward and backward
- Learning and implementing various turns (Pivot Turn, Spin Turn, Arc Turn)
- Observe limitations of robots moving without sensor feedback
- Combine coding and problem-solving skills to navigate your robot to various targets in the challenge level
- Explore extension activities before proceeding to the next activity

External Resources

(to be supplied at a later date pending Activity Guides indicating instructional supports needed)

- Observation log – blank one, and one for the Extension Activity
- Navigation Cheat Sheet
- The Engineering Process

ASSESSMENT

Rubric

Concept	Distinguished	Proficient	Competent	Developing
Robot Design concepts	Can explain robot design components and how they work ,.Project demonstrates use of all motors and pivot points on the robot. Definitions of components are included in student's presentation.	Demonstrates use of motors on the robot. Full understanding of robot components is indicated verbally or in project presentation.	Can make robot move, provides basic understanding of motors. Basic evidence of hardware components understanding is indicated in student's project presentation.	No evidence of understanding robot parts and how they function. No inclusion or mention of hardware components in student's presentation.
Software concepts	Complex command combinations and project goals demonstrated. Can explain command groups needed for using motors, robot balance, and pivot point. Commands or groups of commands are included in student's presentation or documentation.	Student understands command groups needed for using motors, robot balance, and pivot point. Commands or groups of commands mentioned in student's project presentation.	Student has basic understanding of commands needed for using motors, robot balance, and pivot point, but may not be mentioned in student's presentation.	No evidence of understanding commands and how they function. No inclusion or mention of commands in student's presentation.
Coding concepts	Can debug code errors. Complex code combinations and unique use demonstrated. Can explain codes from most sections. Codes are included in student's presentation or documentation.	Demonstrates understanding of default settings, loops, and conditional commands. Student references at least one of each command type code in their project.	Student has basic understanding of codes, but may not be mentioned in student's project presentation.	No evidence of understanding command codes and how they function. No inclusion or mention of codes in student's presentation.
Real-world concepts	Has innovative ideas to create real-world uses of robots, coding, and movement. Demonstrates understanding and documents it.	Understands use of coding, movement, and robot uses in real-world applications. Includes at least one example in student's project presentation.	Basic understanding of coding, movement, and robot in real-world applications. May be explained verbally when prompted; may not be included in presentation.	No evidence of understanding real-world applications using coding, movement, and robots.
Challenge Activity	Able to reach targets in the challenge efficiently by logging each attempt and making informed improvements with each iteration. Can identify multiple different configurations for reaching the same target.	Able to reach multiple targets. Uses logging of attempts to make improvements with each iteration.	Able to reach one or more targets. Limited evidence of logging each attempt or ineffective use of logging to quickly find the target.	Unable to reach any targets. No evidence of logging attempts, lacks understanding how motor settings move the robot.

STANDARDS MAPPING

Common Core Standards

Common Core Standards link

Math skills can be reinforced when considering the distance to the targets, calculating those settings for the movement codes, and testing the results.

CSTA Standards for Students

https://csteachers.org/Page/standards

1B-AP-12

Modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

1B-AP-15

Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.

2-AP-10

Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-13

Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-17

Systematically test and refine programs using a range of test cases.

3A-AP-13

Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-16

Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3A-AP-17

Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-22

Design and develop computational artifacts working in team roles using collaborative tools.

ISTE Standards for Students

ISTE Standards for students link

1.a, 1.b, 1.c, 1.d

Students leverage technology to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.

2.b, 2.d
Students engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices. Students manage their personal data to maintain digital privacy and security and are aware of data-collection technology used to track their navigation online.
3.a, 3.d
Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts, and make meaningful learning experiences for themselves and others.
4.a, 4.b, 4.c, 4.d
Students use a variety of technologies within a design process to identify and solve problems by creating new, useful, or imaginative solutions. This includes creating innovative artifacts or solving authentic problems, considering design constraints and calculated risks, using prototypes as part of a cyclical design process, and developing capacity to work with open-ended problems.
5.a, 5.b, 5.c, 5.d
Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
6.b, 6.c, 6.d
Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats, and digital media appropriate to their goals.
7.b, 7.c
Students use digital tools to broaden their perspectives and enrich their learning by collaborating with others and working effectively in teams locally and globally.

NCSS Standards
NCSS Standards link

NGSS Standards
NGSS Standards link

Interdisciplinary and 21st Century Connections

This lesson covers areas related to Coding/Computer Science.
21st Century Connections: Critical thinking • Creativity • Collaboration • Communication • Technology literacy • Flexibility • Leadership • Initiative • Problem solving • Mathematical calculations • Predictive reasoning

Modifications and Accommodations

Provide modifications and accommodations as appropriate based on student needs, IEP, 504, etc.

- Students can work in teams to integrate a paired programming approach
- Sample code can be provided for students to deconstruct and modify
- Provide adaptive mouse and/or trackball for navigation
- Provide larger monitor for ease of viewing the entire program
- Provide facilitated support to refocus on appropriate panel of software and reinforce nomenclature

LESSON PLAN

LET'S TRAIN VIRTUAL ROBOTS

LESSON 1: ROBOT VEHICLES

UNREAL ENGINE