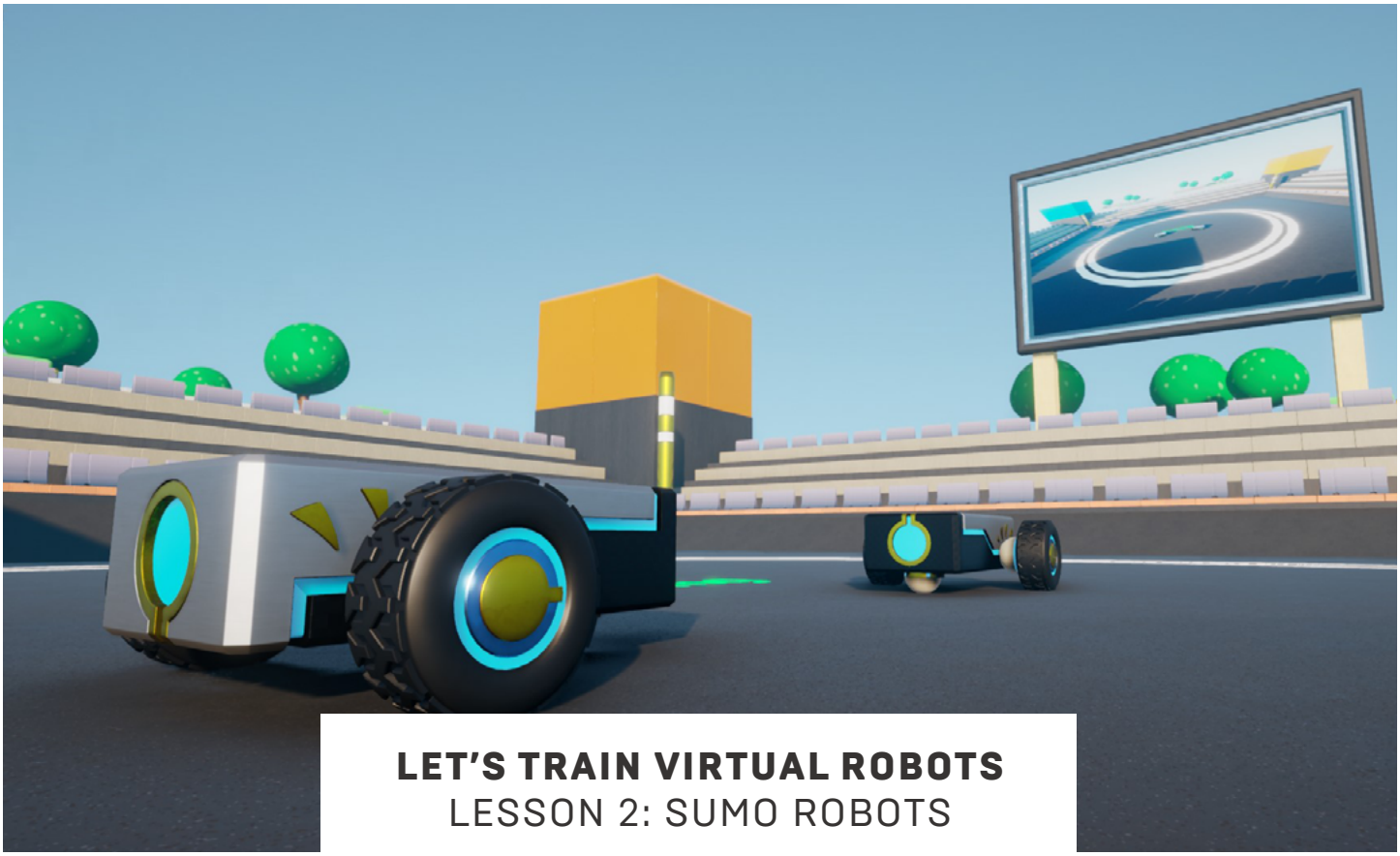
LET'S TRAIN VIRTUAL ROBOTS

LESSON 2: SUMO ROBOTS

UNREAL ENGINE

LESSON PLAN

TABLE OF CONTENTS



LET'S TRAIN VIRTUAL ROBOTS
LESSON 2: SUMO ROBOTS

LESSON CLASS GUIDE INFORMATION

Lesson Title: Let's Train Virtual Robots: **Sumo Robots**
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 robotics learning activity requires the installation of the Unreal Engine software and the Unreal Learning Kit files. The minimum hardware requirements are: arning 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 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 Ian 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 Unreal Learning Kit 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 TWO – SUMO ROBOTS

Introduction

Would you like to make a robotic sumo wrestler? We thought so!

But already there are questions. For instance, how can you help your robot follow the rules without any added input from you? How does it begin to "see" the out-of-bounds borders that help it win or lose a match? You're about to find out!

Activity Overview

This lesson will explain why robots need sensors, how to use one to receive information, and how robots can make decisions based on that information.

Driving robots without any input leads to varied or unreliable results. How can we make driving more accurate and consistent?

Adding sensors provides information that can be used to help a robot decide which particular action must be taken to accomplish a goal.

Understanding the type of information provided by a sensor can help students discern how that information is used to make a decision. The sensor used in this lesson will return a single value to identify the current feedback within a range of possible values. Our lesson will work with a sensor-controlled robot moving to, but not over a line. The robot's ability to detect the line is impacted by reflected light. The student will establish a sensor feedback threshold number to identify when the robot should respond differently. Consider a threshold of 46 for a sensor that returns a range of 0 to 100. Any feedback below 46 will cause one action, while feedback of 46 to 100 will cause a different action. Using a threshold allows us to convert a range of numbers into a Boolean.
Our sandbox for this lesson will be a sumo-bot challenge. If you're new to sumo, the goal is to stay in the circle, as you push your opponent out of it. Let's see how your robots do!

Coding Connection

This activity will introduce the importance of conditional statements. The robot will have a sensor that can detect light and dark. We will test the real-time sensor value against a threshold value to determine if it is seeing the arena or the out-of-bounds line. The conditional statement will allow the robot to perform different actions based on seeing the arena or the out-of-bounds line.

Essential Questions

1. What are the benefits and the challenges for dead-reckoning (navigating without the use of sensor input) versus sensor-enabled driving?
2. How does a robot "see" in this lesson?
   (Answer: with a light sensor detecting the difference between light and dark colors)
3. How will the robot's ability to "see" affect the accuracy of its navigation?
4. What is the sensor's input value of the sumo arena floor, and the out-of-bounds line?
   (Arena floor is white, thus a higher number from the sensor. The black line reports lower numbers.)
5. How do we use the sensor's information to decide when the robot hits the out-of-bounds line?
   (Set a threshold. Any value below is the black line. Any value greater or equal is the arena floor)
6. How does the position of a sensor affect the outcome of its movement and/or ending result?
   (If the sensor is too high, it becomes less effective due to loss of reflected light intensity.)
7. How does the color of the ground affect a reflected light sensor?
   (Lighter surfaces return higher numbers than darker surfaces.)
8. What movement does the robot need to perform based on the sensor feedback?
   (If in the arena, drive forward. If the out-of-bounds line is detected, turn around.)

Learning Outcomes

After completing this lesson, the student will be able to:

1. Recognize the value of movement based on sensor-driven "sight," which helps the accuracy and dependability of a robot
2. Use a light sensor, process settings for input, and control how that information is interpreted
3. Use the current sensor value to determine whether the robot is in the arena or on the out-of-bounds line
4. Assign movement instructions to the robot when it detects the arena floor
5. Assign movement instructions to the robot when it detects the out-of-bounds line
6. Use the Blueprint programming language to implement a loop and conditional statements that make the robot constantly look for a line on the ground/surface

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 this 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

Defining a Purpose

We are creating a robot with a specific purpose. Because we know it'll be a robot sumo, we can make assumptions about the environment, set rules to follow, and define some basic requirements for it.

Environment

- The robot starts inside a flat circular arena
- The arena has an identifiable out-of-bounds line

Rules

- The robot is always moving
- The robot tries to stay inside the arena circle

Requirements

- Uses a sensor to "see" the ground
- Drives straight when it "sees" the arena floor
- Turns around when it "sees" the out-of-bounds line

ACTIVITIES

Moving the Robot

Students will use Blueprint node-based coding to control the motor resulting in the movement of the robot.

Activity: Add robot to the arena and make it drive forward

Adding a Sensor

Since the robot needs to detect the difference between the arena floor and the out of bounds line, we will add a sensor that can help detect the difference.

Sensor Type

- We will use a Light Sensor that will measure reflected light. This gives a higher value on a light surface and a lower value on a dark surface

Sensor Location

- The sensor must look down at the ground while the vehicle is driving
- The sensor must be close to the ground to get an accurate reading
- The sensor should be located near the front middle of the robot to avoid driving out of bounds

Activity: Add a Light Sensor to the robot and compare the values when seeing the arena vs. the out-of-bounds line. Note the values and establish a threshold value that divides the full range of sensor feedback into “seeing the arena floor” and “seeing the out-of-bounds line”

Getting Input from a Sensor

Once the sensor is physically connected to the robot in an ideal location, it will need to be added to the robot’s code.

Activity: Edit the robot blueprint to add the Run Sensor component for the Light Sensor.

Responding to Sensor Feedback

When the robot is able to read the input values from the sensor, it can perform a specific action based on the feedback from the sensor.

- If it sees the arena floor, drive forward
- If it sees the out-of-bounds line, turn around

Activity: Add a conditional statement (Branch) in the blueprint code to perform the actions based on the sensor input. If the sensor input is lower than the threshold, it is seeing the out-of-bounds line. If the input value is higher than the threshold, it is seeing the arena floor.

Activity: Add the code necessary for the robot to turn around when seeing the out-of-bounds line. Consider the specific actions that will be assigned to each motor to complete the turn-around maneuver.

Challenge: Code the robot to stay inside the sumo arena without any user intervention.

Sumo Battle

Now it's time to put your work to the test. We will introduce a duplicate version of your working robot and observe as they battle each other.

Activity: Duplicate your robot and battle them against each other in the arena. How do they perform? What changes could improve your robot?

CHALLENGE ACTIVITY

Extension Activities

- Modify one of your robots to help improve its' success rate
- Import robots from other students to see how your robot compares

Activity Review

- Understanding the need for sensors, and how they analyze input.
- Understanding sensor effectiveness when placed in different positions on the robot.
- Understanding the use of a threshold to convert a range of sensor values to a Boolean.
- Understanding coding commands to employ sensors to determine robot movement.
- Coding connection: Conditional Statement using the threshold test. Using Loops to repeat actions.

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
- Threshold Calculation Worksheet

ASSESSMENT

Rubric

Concept	Distinguished	Proficient	Competent	Developing
Robot Design concepts	Can explain robot sensor components and how they work to others.Project demonstrates use of sensors in lesson, and the best positions on the robot. Definitions of components are exhibited, by demonstration or in student documentation.	Demonstrates use of sensors on the robot. Full understanding of robot components is indicated to teacher.	Can make robot respond to sensors, but does not provide meaningful understanding of sensors. Basic understanding of hardware components demonstrated.	No evidence of understanding robot sensors and how they function. No understanding of hardware components demonstrated.
Software concepts	Complex command combinations and project goals demonstrated. Can explain command groups needed for using sensors, sensor position, and desired resulting robot position or action. Commands or groups of commands are demonstrated by presentation or documentation.	Student understands command groups needed for using sensors, sensor position, and desired resulting robot position or action. Commands or groups of commands mentioned in student's presentation.	When prompted, student has basic understanding of commands needed for using sensors, sensor position and desired resulting robot position or action, but may not be presented in demonstration.	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 demonstration or documentation.	Demonstrates understanding of default settings, loops, and conditional commands. Student references at least one of each command type code in their presentation.	When prompted, student has basic understanding of codes, but may not be mentioned in student's presentation.	No evidence of 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 some use of coding, movement, and robot uses in real-world applications. Includes at least one example in student's presentation.	Basic understanding of existence of coding, movement, and robot in real-world applications. May be explained verbally when prompted, but may not be included in presentation.	No evidence of understanding real-world applications using coding, movement, and robots.
Challenge Activity	Demonstrates innovative ideas or additional tools added to the sumo bot model and actions. Was able to replicate the robot to battle another robot.	Was successful in coding the robot to move within the sumo arena and turn around when it detects the out of bounds line.	Basic understanding of how to code a sumo bot, but was not able to complete the sumo-bot actions to stay within the arena.	Does not understand the functions of a sumo-bot, did not demonstrate how sensors function, and did not code sumo-bot activity.

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-10

Create programs that include sequences, events, loops, and conditionals.

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 focus on appropriate software panels and reinforce nomenclature
- Add text labels next to colors, when working with the light or color sensors
- Add alternative measuring labels when working with the ultrasonic sensor's thresholds

LESSON PLAN

LET'S TRAIN VIRTUAL ROBOTS

LESSON 2: SUMO ROBOTS

UNREAL ENGINE