UNIVERSITY TRANSPORTATION CENTER FOR

RAILWAY SAFETY

# University Transportation Center for Railway Safety (UTCRS)

## HIGH SCHOOL STEM CURRICULUM








## Table of Contents

2017 Next Generation Science Standards 9-12 ..... 3
International Technology and Engineering Educator Association (ITEEA) Standards ..... 3
Physics Texas Essential Knowledge and Skills (TEKS) ..... 4
Math Texas Essential Knowledge and Skills (TEKS) ..... 5
**Day 1: Introduction to Python** ..... 6
Activity 1: Team Building – Logic Labyrinth Challenge ..... 7
Activity 2: Introduction to Variables ..... 8
Activity 3: Understanding Loops and Logic ..... 10
Activity 4: Programming Hardware – Traffic Lights ..... 11
**Day 2: Sensor Programming** ..... 13
Activity 1: Understanding the Sensors ..... 15
Activity 2: Simulating a Railroad Intersection (Active Sign Creation) ..... 16
Activity 3: Rover Build and Navigation ("Alex" Robot) ..... 18
Activity 4: Rover Speed and Acceleration ..... 19
**Day 3: Hardware Applications** ..... 21
Activity 1: Rover Race ..... 23
Activity 2: Line Follower (Rover Vision) ..... 25
Activity 3: Collision Simulation ..... 26
Activity 4: Forklift Assembly Challenge ..... 27
**Day 4: Final Challenge** ..... 29
Activity 1: Final Challenge ..... 30
**Appendices** ..... 33
Appendix A: Video Links ..... 33
Appendix B: Teacher Notes ..... 34
**References** ..... 35
**Curriculum Writers** ..... 37




# Learning Objectives

## 2017 Next Generation Science Standards 9-12

| | |
|---|---|
| HS-PS2-1 | Analyze data to support the claim that Newton's second law of motion describes the mathematical relationship among the net force on a macroscopic object, its mass, and its acceleration. |
| HS-PS3-3 | Design, build, and refine a device that works within given constraints to convert one form of energy into another form of energy. |
| HS-PS4-1 | Use mathematical representations to support a claim regarding relationships among the frequency, wavelength, and speed of waves traveling in various media. |
| HS-PS4-5 | Communicate technical information about how some technological devices use the principles of wave behavior and wave interactions with matter to transmit and capture information and energy. |

## International Technology and Engineering Educator Association (ITEEA) Standards

| | |
|---|---|
| STEL-1N | Explain how the world around them guides technological development and engineering design. |
| STEL-2T | Demonstrate the use of conceptual, graphical, virtual, mathematical, and physical modeling to identify conflicting considerations before the entire system is developed and to aid in design decision making. |
| STEL-2Z | Use management processes in planning, organizing, and controlling work. |
| STEL-3H | Analyze how technology transfer occurs when a user applies an existing innovation developed for one function to a different purpose. |
| STEL-4P | Evaluate ways that technology can impact individuals, society, and the environment. |
| STEL-5H | Evaluate a technological innovation that arose from a specific society's unique need or want. |
| STEL-6F | Relate how technological development has been evolutionary, often the result of a series of refinements to basic inventions or technological knowledge. |
| STEL-7Y | Optimize a design by addressing desired qualities within criteria and constraints. |
| STEL-8P | Apply appropriate methods to diagnose, adjust, and repair systems to ensure precise, safe, and proper functionality. |

## Physics Texas Essential Knowledge and Skills (TEKS)

| Process Standards |
|---|

| | |
|---|---|
| P.P.1 | asks questions, identifies problems, and plans and safely conducts classroom, laboratory, and field investigations to answer questions, explain phenomena, or design solutions using appropriate tools and models.<br>P.1(B) apply scientific practices to plan and conduct descriptive, comparative, and experimental investigations and use engineering practices to design solutions to problems |
| P.P.2 | analyzes and interprets data to derive meaning, identify features and patterns, and discover relationships or correlations to develop evidence-based arguments or evaluate designs.<br>P.2(D) evaluate experimental and engineering designs |
| **Readiness and Supporting Standards** | |
| P.5(A) | analyze different types of motion by generating and interpreting position versus time, velocity versus time, and acceleration versus time using hand graphing and real-time technology such as motion detectors, photogates, or digital applications |
| P.5(B) | define scalar and vector quantities related to one- and two-dimensional motion and combine vectors using both graphical vector addition and the Pythagorean theorem |
| P.5(D) | describe and analyze acceleration in uniform circular and horizontal projectile motion in two dimensions using equations |
| P.5(E) | explain and apply the concepts of equilibrium and inertia as represented by Newton's first law of motion using relevant real-world examples such as rockets, satellites, and automobile safety devices |
| P.5(F) | calculate the effect of forces on objects, including tension, friction, normal, gravity, centripetal, and applied forces, using free body diagrams and the relationship between force and acceleration as represented by Newton's second law of motion |
| P.5(G) | illustrate and analyze the simultaneous forces between two objects as represented in |
| P.6(B) | identify and describe examples of electric and magnetic forces and fields in everyday life such as generators, motors, and transformers |
| P.6(D) | analyze, design, and construct series and parallel circuits using schematics and materials such as switches, wires, resistors, lightbulbs, batteries, voltmeters, and ammeters |
| P.7(A) | calculate and explain work and power in one dimension and identify when work is and is not being done by or on a system |
| P.7(D) | calculate and describe the impulse and momentum of objects in physical systems such as automobile safety features, athletics, and rockets |
| P.8(E) | compare the different applications of the electromagnetic spectrum, including radio telescopes, microwaves, and x-rays |




# Math Texas Essential Knowledge and Skills (TEKS)

| Content and Process Standards | |
|---|---|
| 2A.P.1(A)<br>G.P.1(A) | apply mathematics to problems arising in everyday life, society, and the workplace |
| A.P.1(B)<br>2A.P.1(B)<br>G.P.1(B) | use a problem-solving model that incorporates analyzing given information, formulating a plan or strategy, determining a solution, justifying the solution, and evaluating the problem-solving process and the reasonableness of the solution |
| A.P.1(C)<br>2A.P.1(C)<br>G.P.1(C) | select tools, including real objects, manipulatives, paper and pencil, and technology as appropriate, and techniques, including mental math, estimation, and number sense as appropriate, to solve problems |
| A.P.1(D) | communicate mathematical ideas, reasoning, and their implications using multiple representations, including symbols, diagrams, graphs, and language as appropriate |
| A.P.1(G) | display, explain, and justify mathematical ideas and arguments using precise mathematical language in written or oral communication |
| **Readiness and Supporting Standards** | |
| **Algebra 1[A]** | |
| A.2(A) | determine the domain and range of a linear function in mathematical problems; determine reasonable domain and range values for real-world situations, both continuous and discrete; and represent domain and range using inequalities |
| A.12(B) | evaluate functions, expressed in function notation, given one or more elements in their domains |
| A.12(C) | identify terms of arithmetic and geometric sequences when the sequences are given in function form using recursive processes |
| **Algebra 2 [2A]** | |
| 2A.3(B) | solve systems of three linear equations in three variables by using Gaussian elimination, technology with matrices, and substitution |
| 2A.7(I) | write the domain and range of a function in interval notation, inequalities, and set notation |
| 2A.8(C) | predict and make decisions and critical judgments from a given set of data using linear, quadratic, and exponential models |
| **Geometry [G]** | |
| G.4(A) | distinguish between undefined terms, definitions, postulates, conjectures, and theorems |
| G.4(B) | identify and determine the validity of the converse, inverse, contrapositive of a conditional statement and recognize the connection between a biconditional statement and a true conditional statement with a true converse |
| G.4(C) | verify that a conjecture is false using a counterexample |
| G.9(B) | apply the relationships in special right triangles 30°-60°-90° and 45°-45°-90° and the Pythagorean theorem, including Pythagorean triples, to solve problems |

# Day 1: Introduction to Python




**Theme:** Programming Principles, Logic, and Team Collaboration

**Day Overview:** Students gain an understanding of how engineers collaborate in teams to achieve common goals. They are introduced to Python programming basics, including variables, functions, logic statements, and loops. Using Pi-top [4] robotics accessories such as LEDs, buzzers, and buttons, students program interactive tasks and apply logical thinking. The day emphasizes teamwork, problem-solving, and applying coding concepts to real-world scenarios.

**Lesson Objectives:**

By the end of Day 1, students will be able to:

- Demonstrate effective teamwork and communication when solving problems in interdisciplinary teams.
- Create Python programs using variables, functions, and logic statements.
- Apply loops to simplify repetitive programming tasks.
- Interface Pi-top [4] hardware components (LEDs, buttons, buzzers) with Python code.
- Build and execute programs that model real-world processes like traffic signals.
- Critically reflect on problem-solving strategies and iterative improvement.

## Day 1 at a Glance

| Time Frame | Activity | Focus / Goal |
|---|---|---|
| 30–40 min | Team Building – Logic Labyrinth Challenge | Develop teamwork, communication, and algorithmic thinking by guiding a blindfolded "computer" through a maze. |
| 45 min | Introduction to Variables | Learn to define, assign, and manipulate variables; import libraries and control Pi-top [4] accessories. |
| 60–75 min | Understanding Loops and Logic | Use conditional and relational statements to create loops and algorithms; program an interactive counter on the Mini Screen. |
| 100 min | Programming Hardware – Traffic Lights | Build and code a traffic light system using LEDs, buttons, and buzzers; define functions and handle multiple events. |

## Day 1 Vocabulary

| Term | Student-Friendly Definition | Day 1 Connections |
|---|---|---|
| Scalar | A measurement that only has an amount, not a direction. | On Day 1, students use scalar variables to set baseline values like LED delays. |
| Vector | A measurement that includes both an amount and a direction. | On Day 1, students give specific vector-like directions to guide blindfolded teammates. |
| Velocity | The speed of an object in a specific direction. | On Day 1, students map simple turning paths. |
| Photoresistor | A light sensor component that changes electrical resistance with brightness. | On Day 1, students build variables to turn hardware on and off. |

## Activity 1: Team Building – Logic Labyrinth Challenge

**Estimated Time:** 30–40 minutes

**NGSS Connections:** MS-ETS1-1, MS-ETS1-2

**TEKS Connections:** A.P.1(B)

**Activity Overview**

Students work in small groups to solve a maze challenge using programming logic. One student is blindfolded ("computer") while the rest ("programmers") devise an algorithm to guide them safely through the maze without collisions. The activity emphasizes collaboration, communication, and algorithmic thinking.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|

| | | | |
|---|---|---|---|
| P.P.2, MS-ETS1-1 | Programming Logic & Engineering Design | Algorithms always work the first time; steps are obvious | Students may give incomplete or incorrect instructions and expect the "computer" to navigate flawlessly |
| MS-ETS1-2 | Team Problem Solving | Teamwork is optional | Students may try to solve the maze individually and fail |
| MS-ETS1-1 | Planning Investigations | Steps can be skipped | Students may not follow a structured algorithm, causing collisions |

**Materials Needed**

- Maze (tape on floor or physical barriers)
- Pen and paper
- Blindfold
- Timer

**Activity Flow**

1. Prep the maze with multiple pathways.
2. Divide students into teams of 3–4.
3. Assign roles: one "computer" and programmers.
4. Students write step-by-step instructions to guide the blindfolded "computer".
5. Execute the algorithm, adding penalties for collisions.
6. Record completion times and discuss strategies.

**Reflection Questions:**

1. What was challenging, and how did you address those challenges?
2. How important was communication in your team?
3. What would you do differently to improve results?

---

## Activity 2: Introduction to Variables

**Estimated Time:** 45 minutes

**NGSS Connections:** MS-ETS1-1, MS-ETS1-2

**TEKS Connections:** P.P.2, A.P.1(G), A.12(B)




**Activity Overview**

Students learn to define and assign variables in Python and manipulate them through the Further website. They import libraries to control Pi-top [4] hardware like LEDs and buzzers.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| P.P.2, MS-ETS1-1 | Variables & Assignment | Variables automatically update or perform calculations without explicit instructions | Students may expect an LED or counter to change without coding proper logic |
| A.P.1(G) | Library Imports | Libraries are optional or unnecessary | Students may not realize that importing Pi-top SDK functions is required for hardware control |
| A.12(B), MS-ETS1-2 | Programming Practices | Typos or syntax errors are trivial | Students may be confused when code doesn't run due to incorrect variable names or improper Python syntax such as incorrect indentations or typos |

**Materials Needed:**

- Pi-top [4] and Expansion Plate
- Laptop or smart device with Internet access
- LEDs

**Activity Flow:**

1. Turn on the Pi-top [4] and connect to a smart device.
2. Sign up for Further (https://www.further.pi-top.com).(You will need to access Further for the remainder of these activities).
3. Open "Python Principles: Intro to Variables" and follow guided instructions.
4. Connect LEDs to Pi-top [4] and manipulate variables to control them.
5. Optionally explore "Python Principles: The Import System."

**Reflection Questions:**

1. Why should programmers be specific when writing code?
2. How do import statements work?
3. How can we teach the computer what each Pi-top accessory is and how to control it?

# Activity 3: Understanding Loops and Logic

**Estimated Time:** 60–75 minutes

**NGSS Connections:** MS-ETS1-3, MS-ETS1-4

**TEKS Connections:** P.P.1, P.P.2, A.2(A), A.12(C), 2A.7(I), 2A.3(B), G.4(A), G.4(B)

**Activity Overview:**
Students create complex algorithms using conditional statements and loops to program an interactive counter on the Pi-top [4] Mini Screen. They practice simplifying repetitive tasks and understanding logical flow.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| P.P.2, MS-ETS1-3 | Loops & Conditional Logic | Loops are unnecessary; if statements always behave predictably | Students may misuse "if" or "for" statements and expect sensors to behave without proper logic |
| A.2(A) | Algorithm Design | Repetition is inefficient | Students may write long repetitive code instead of using loops |
| G.4(A), G.4(B), MS-ETS1-4 | Computational Thinking | Sensors automatically interact correctly | Students may assume the Mini Screen or LED responds correctly without proper programming |

**Materials Needed:**

- Pi-top [4] and Expansion Plate
- Laptop with Internet access
- Button, LEDs

**Activity Flow**

1. Access "Understanding Logic" course on Further.
2. Complete "If Statements," "Loops," and "Tutorial: Mini Screen" lessons.
3. Program an interactive counter using loops and conditional logic.

**Reflection Questions:**

1. What is an "if statement," and why is it used?

2. Describe what happens when a condition is true vs. false.
3. How do different loops differ, and why use them with sensors?

---

## Activity 4: Programming Hardware – Traffic Lights

**Estimated Time:** 100 minutes

**NGSS Connections:** MS-ETS1-3, MS-ETS1-4

**TEKS Connections:** P.6(D), P.8(E), A.P.1(A), A.P.1(B), A.P.1(C), A.P.1(G), A.2(A), 2A.7(I), A.12(B), A.12(C), 2A.3(B), 2A.8(C), G.9(B)

**Activity Overview**
Students program traffic light sequences using Pi-top [4] hardware, defining functions and handling multiple events simultaneously. This reinforces loops, event handling, and real-world application of programming logic.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| A.P.1(A-C, G), MS-ETS1-3 | Functions & Event Handling | Functions automatically handle multiple events; event handling is optional | Students may expect LEDs and buzzers to react correctly without proper function definition |
| A.2(A), MS-ETS1-4 | Algorithm Design | Complex sequences are unnecessary | Students may try to control traffic lights manually instead of using loops |
| 2A.7(I), 2A.8(C) | Computational Thinking | Loops are optional | Students may code sequential commands without using loops |
| P.6(D), P.8(E) | Hardware Integration | LEDs/buttons don't require code | Students may assume traffic lights function automatically without code |

**Materials Needed:**

- Pi-top [4] with Expansion Plate
- Pi-top [4] Robotics Kit
- Laptop with Internet access
- LEDs, Buttons, Buzzer




**Activity Flow:**

1. Watch “How Do Traffic Signals Work” video.
2. Access Further and open “Programming Hardware” course, Traffic Lights lesson.
3. Follow instructions to program LEDs, buttons, and buzzer with functions and loops.

**Reflection Questions:**

1. How do traffic signals improve safety?
2. How does your model compare to real-world systems?
3. Why are loops important for programs mimicking safety signals?




# Day 2: Sensor Programming

**Theme:** Applying Python and Sensor Technology for Railroad Safety

**Day Overview:** Students will explore how Python programming can control various sensors, including light, sound, and ultrasonic sensors. They will design and build a railroad intersection, integrating traffic signals and servo motors, and construct the "Alex" robot while programming its movement. The day emphasizes coding, sensor integration, and teamwork, reinforcing real-world applications of technology in railway safety systems.

**Lesson Objectives:**

By the end of Day 2, students will be able to:

- Apply Python programming to control light, sound, and ultrasonic sensors.
- Design and implement traffic signals at a railroad intersection using servo motors and LEDs.
- Build and program the "Alex" robot to move forward, backward, and turn using code.
- Combine hardware and software to simulate real-world railway safety systems.
- Collaborate effectively to plan, test, and refine robotic and sensor-based solutions.
- Determine and calculate speed and acceleration of a robot using Python and encoder data.
- Apply loops, functions, and iterative statements to control robot motion.
- Analyze motion data with graphs and compare results to theoretical predictions.

## Day 2 at a Glance

| Time Frame | Activity | Focus / Goal |
|---|---|---|
| 30–45 min | Understanding the Sensors | Learn how light, sound, and ultrasonic sensors work; collect data using Python; explore sensor limitations |
| 90 min | Simulating a Railroad Intersection (Active Sign Creation) | Apply Python and hardware integration to design active railroad crossing signs; understand the difference between passive and active signals |
| 120 min | Rover Build and Navigation ("Alex" Robot) | Build and program a robot to follow commands; understand hardware-software integration and motion control |
| 60 min | Rover Speed and Acceleration | Measure velocity, acceleration, and graph motion data; apply loops and Python calculations. |

| | | |
|---|---|---|
| 15 min | Reflection & Debrief | Discuss successes, challenges, and real-world applications of sensors, robotics, and railroad safety systems |

**Day 2 Vocabulary**

| Term | Student-Friendly Definition | Day 2 Connections |
|---|---|---|
| Scalar | A measurement that only has an amount, not a direction. | On Day 2, students will analyze numerical motor values to program vehicle operations. |
| Vector | A measurement that includes both an amount and a direction. | On Day 2, students will program the robot to navigate paths and track coordinates. |
| Speed | How fast an object moves (Speed = Distance / Time). | On Day 2, students write Python algorithms to measure the robot's real-time velocity, which helps them optimize remote steering control during the Day 4 obstacle challenge. |
| Velocity | The speed of an object in a specific direction. | On Day 1, students mapped simple turning paths, which builds up to Day 2 scripts where the robot adjusts steering velocity to follow complex track patterns. |
| Acceleration | How quickly speed or direction changes over time. | On Day 2, students compute changes in motor speeds using internal wheel encoders, preparing them to manage weight shifts while hauling payloads on Day 4. |
| Frequency | The number of wave cycles that pass in one second. | On Day 2, students explore wave signals to program audio alerts, which directly relates to the ultrasonic tracking sweeps used during the Day 4 navigation run. |
| Wavelength | The distance between two identical, consecutive peaks of a wave. | On Day 2, students study wave structures to understand how different sensor hardware |

| | | |
|---|---|---|
| | | intercepts information from the surrounding room environment. |
| Amplitude | The maximum height or strength of a wave from rest. | On Day 2, students analyze changes in sound pressure levels to write threshold code for their hardware sensors. |
| Photoresistor | A light sensor component that changes electrical resistance with brightness. | On Day 1, the students built variables to turn hardware on and off, while on Day 2, they will use light tracking sensors to detect visual changes along the floor path. |

## Activity 1: Understanding the Sensors

**Estimated Time:** 30–45 minutes

**NGSS Connections:** HS-PS4-1, HS-PS4-5

**TEKS Connections:** P.P.2, P.6(D), P.8(E), A.P.1(A), A.P.1(B), A.P.1(C), A.P.1(G), 2A.3(B), 2A.8(C), G.9(B)

**Activity Overview:**
Students learn how light, sound, and ultrasonic sensors collect data and interact with Python code. They will experiment with each sensor, including measuring distance with the ultrasonic sensor and detecting light or sound changes, while understanding limitations of each sensor.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS4-1 | Waves & Signals | Sensors automatically detect everything accurately | Students may expect the ultrasonic sensor to measure distances through all materials |
| HS-PS4-5 | Information Transfer | All sensors detect signals in the same way | Students may assume the light sensor works like a sound sensor or vice versa |
| P.P.2 | Programming Practices | Code alone controls the sensor; hardware doesn't matter | Students may overlook sensor placement and environment effects |

| | | | |
|---|---|---|---|
| A.P.1(G) | Algorithm Design | One approach works in all scenarios | Students may assume a single code snippet works for every sensor situation |

**Materials Needed:**

- Pi-top [4]
- Expansion Plate
- Laptop with Internet Access
- Light Sensor
- Sound Sensor
- Ultrasonic Sensor
- Flashlight
- Acoustic Foam (optional)

**Activity Flow:**

1. Show videos explaining how each sensor works (sound, ultrasonic, light).
2. Open Further website and navigate to the "Understanding the Sensors" course.
3. Guide students through each sensor lesson.
4. Demonstrate limitations, e.g., how acoustic foam interferes with ultrasonic sensors.

**Reflection Questions:**

1. How can data be collected from a sound sensor using code?
2. How can a sensor detect the distance to an object using code?
3. How can code be modified to display data in different units?
4. Why are events used in programming?
5. What program would alert a user if a room is too bright?

---

## Activity 2: Simulating a Railroad Intersection (Active Sign Creation)

**Estimated Time:** 90 minutes

**NGSS Connections:** HS-PS4-5

**TEKS Connections:** P.P.1, P.P.2, P.5(E), P.6(B), P.6(D), P.7(D), P.8(E), 2A.P.1(A), A.P.1(B), A.P.1(D), A.P.1(G)




**Activity Overview**

Students will integrate servo motors, LEDs, and buzzers to create active railroad crossing signs that respond to a simulated train. This activity teaches programming logic, hardware control, and the importance of active safety signals.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS4-5 | Information Transfer | All railway signals are the same | Students may confuse passive signs with active signs |
| P.P.2 | Programming Practices | Servo motors function without coding | Students may place gates but forget to write code for activation |
| A.P.1(B) | Algorithm Design | One code block works for every sensor | Students may write a code that works for the light sensor but not the ultrasonic sensor |
| P.5(E) | Hardware Integration | Hardware is plug-and-play | Students may overlook wiring or component orientation |

**Materials Needed:**

- Pi-top [4]
- Expansion Plate
- Pi-top [4] Robotics Kit
- Laptop
- 2 Servo Motors
- 2 Red LED Lights
- 1 Buzzer
- 1 Ultrasonic Sensor

**Activity Flow:**

1. Discuss the difference between passive and active railway crossing signs.
2. Demonstrate flashing red lights, gates, and bells.
3. Students design and program an active railroad crossing using their components.

**Reflection Questions:**

1. Which signs at a railroad crossing are active vs. passive?
2. How do these signals improve safety?
3. What sensors could enhance the crossing design further?

# Activity 3: Rover Build and Navigation ("Alex" Robot)

**Estimated Time:** 120 minutes

**NGSS Connections:** HS-PS3-3

**TEKS Connections:** P.5(D), P.6(D), G.9(B)

**Activity Overview**

Students will construct the "Alex" robot and program it to move forward, backward, and turn using Python. They learn how hardware and software work together to create intelligent behavior and control motion.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS3-3 | Energy Transfer | Motors and wheels move automatically | Students may expect the robot to move without coding commands |
| P.5(D) | Robotics Programming | One algorithm fits all navigation tasks | Students may not account for left/right turns or obstacles |
| P.6(D) | Hardware/Software Integration | Software is separate from hardware | Students may not understand how motors respond to Python code |
| G.9(B) | Algorithm Testing | Code works perfectly the first time | Students may not test and iterate robot movement |

**Materials Needed:**

- Pi-top [4]
- Expansion Plate
- Pi-top [4] Robotics Kit
- Laptop with Internet Access

**Activity Flow:**

1. Students build the "Alex" robot using provided instructions.
2. Introduce the drive controller for navigation.
3. Students write Python code to move the robot forward, backward, left, and right.
4. Practice adjusting code for distance, speed, and turning accuracy.

**Reflection Questions:**

1. How do hardware and software combine to collect and exchange data?
2. How is intelligent behavior conveyed through robotics?
3. How can negative values affect robot motion?
4. What are the pros and cons of using sleep commands to control movement?

## Activity 4: Rover Speed and Acceleration

**Estimated Time:** 60 minutes

**NGSS Connections:** HS-PS3-3

**TEKS Connections:** P.5(A), P.5(D), P.5(E), P.5(F), P.5(G), P.6(B), P.6(D), P.7(D), P.8(E), A.P.1(D), A.12(C), G.9(B)

**Activity Overview:**
Students use Python to calculate the rover's speed and acceleration based on encoder motor data. They generate graphs automatically with matplotlib and compare experimental results to hypotheses, learning the effects of mass, speed, and motion on robot performance.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS3-3 | Energy & Motion | Acceleration is only affected by speed, not mass | Students may assume the rover accelerates the same regardless of weight |
| P.5(A), P.5(D), P.5(E) | Using iterative statements for calculations | Loops automatically handle units or conversions | Students may miscalculate graphs if units aren't explicitly included |
| A.P.1(D), A.12(C) | Functions & Graphing | Graphing is optional and not connected to understanding motion | Students may skip labeling graphs, leading to misinterpretation |

**Materials Needed:**

- Pi-top [4]
- Expansion Plate
- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- "Alex" Robot Model

**Activity Flow:**

1. Navigate to Further class and open "Rover Speed and Acceleration" lesson.
2. Explain speed = distance/time and acceleration = speed change/time.
3. Explain the difference between speed and velocity, and that this experiment will only measure speed.
4. Students run the scripts test_speed.py instruct to attach 2 potentiometers and 1 button as indicated in the script and move them so the robot can drive in a straight line.
5. Students must record the highest possible RPM for each of the motors (when it drives in a straight line) and include them in the main.py script, as well as complete the calculate_speed() and calculate acceleration() functions.
6. Generate distance-time and speed-time graphs and add units.
7. Run experiments and compare results to hypotheses and discuss variations.

**Reflection Questions:**

1. Did the data in your graph support your initial hypothesis? Why or why not?
2. How does changing the speed factor over time affect the distance-time graph?
3. What would a deceleration graph look like?
4. How does mass affect speed? What is the relationship between mass and speed?




# Day 3: Hardware Applications

**Theme:** Exploring Computer Vision, Force, Energy, and Engineering Design Through Robotics

**Day Overview:** Students will apply Python programming and robotics to simulate a "Mini Baja Race", simulate collisions, calculate acceleration, force, work, energy, and power, and collaborate to design and program a forklift to manipulate a wheel-axle assembly. The day emphasizes the connection between coding, physics, and hands-on engineering design.

**Lesson Objectives**

By the end of Day 4, students will be able to:

1. Design, program, and control the robot for a simulated race using sensors.
2. Integrate a camera and computer vision functions to create a line follower program.
3. Apply problem-solving, coding logic, and engineering design skills to optimize robot performance.
4. Understand the relationship between force, work, energy, and power.
5. Calculate acceleration, force, work, and power using experimental data.
6. Simulate collisions using their Pi-top [4] rover and interpret results.
7. Design and build a forklift attachment for the robot using the Engineering Design Process.
8. Program the forklift to pick up and drop off a wheel-axle assembly.
9. Analyze and refine designs through testing and iteration.

## Day 3 at a Glance

| Time Frame | Activity | Focus / Goal |
|---|---|---|
| 60 min | Rover Race | Apply coding and iterative statements to control the robot in a race challenge; refine designs. |
| 120 min | Line Follower (Rover Vision) | Learn computer vision basics; implement line-following code and integrate camera functions. |
| 10 min | Welcome & Safety Review | Establish safe lab practices |
| 90 min | Collision Simulation | Calculate acceleration, force, work, and power using rover collisions; graph data |
| 15 min | Break | Refresh and discuss observations from collisions |

| | | |
|---|---|---|
| 90 min | Forklift Assembly Challenge | Design, build, and program a forklift to lift and move a wheel-axle assembly |

## Day 3 Vocabulary

| Term | Student-Friendly Definition | Day 3 Connections |
|---|---|---|
| Vector | A measurement that includes both an amount and a direction. | On Day 1, students give specific vector-like directions to guide blindfolded teammates, while on Day 3 they program the robot to navigate paths and track coordinates. |
| Velocity | The speed of an object in a specific direction. | On Day 1, students map simple turning paths, which builds up to Day 2 and 3 scripts where the robot adjusts steering velocity to follow complex track patterns. |
| Force | A push or a pull (Force = Mass x Acceleration). | On Day 3, students run crash tests to measure how a robot's mass alters collision impacts, which tells them how much motor strain the forklift arm handles on Day 4. |
| Inertia | An object's resistance to changing its current motion state. | On Day 3, students study inertia to see why heavy objects resist moving, allowing them to adjust brake timing when piloting the payload on Day 4. |
| Equilibrium | A balanced state where all combined forces equal zero. | On Day 3, students use balance concepts to keep the vehicle stable, helping them secure heavy components on the forklift without tipping the rover on Day 4. |
| Friction | A resistive force created when two objects slide together. | On Day 3, students analyze push resistance across floor surfaces, which teaches them how to preserve wheel grip and traction during the Day 4 time trial. |
| Tension | A pulling force that passes tightly through a rope or cable. | On Day 3, students map mechanical loads, which helps them construct secure forklift brackets that will not snap under tension on Day 4. |

| | | |
|---|---|---|
| Normal Force | The perpendicular upward support force provided by a solid surface. | On Day 3, students calculate surface support values to understand floor pressure, which helps them balance the robot when lifting heavy payloads on Day 4. |
| Centripetal Force | An inward force keeping an object moving along a circle. | On Day 3, students use circular physics laws to keep turning movements steady, stopping the robot from spinning out of bounds during fast races. |
| Momentum | The quantity of motion an object has based on mass. | On Day 3, students track momentum transfers during car impact scenarios, showing them how cargo weight alters drift during the final Day 4 runs. |
| Impulse | The overall change in momentum caused by a timed force. | On Day 3, students evaluate how impact pads extend crash times to reduce structure damage, which connects to railway bumper engineering principles. |
| Kinetic Energy | The dynamic energy a physical object has because of motion. | On Day 3, students calculate energy transfers from a moving vehicle into a stationary box, teaching them how to control speeds to avoid high-energy wall crashes on Day 4. |
| Work | The energy transferred when a force moves an object. | On Day 3, students apply displacement formulas to map motor outputs, which helps them configure the lifting mechanics of the forklift attachments on Day 4. |
| Power | The speed or rate at which work gets done. | On Day 3, students program real-time power charts using data scripts, allowing them to monitor servo strain limits when lifting heavy rail axles on Day 4. |

## Activity 1: Rover Race

**Estimated Time:** 60 minutes

**NGSS Connections:** HS-PS3-3

**TEKS Connections:** P.5(A), P.5(D), P.5(E), P.5(F), P.5(G), P.6(B), P.6(D), P.7(D), P.8(E), A.P.1(B), A.P.1(D), A.P.1(G), A.12(C), 2A.8(C), G.9(B)




**Activity Overview:**

Students implement loops and iterative statements to control robot motion in a race challenge. They optimize robot performance by applying the Engineering Design Process and prior coding lessons.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS3-3 | Motion & Control | Loops always execute perfectly without debugging | Students may assume robot responds instantly, ignoring lag or code errors |
| A.P.1(B), A.P.1(D), A.P.1(G) | Iterative programming | Adding more keys doesn't affect robot performance | Students may struggle with keyboard polling loops and delays |
| P.5(D), P.6(D) | Engineering Design | Design adjustments are unnecessary | Students may not refine the robot design, leading to slower race times |

**Materials Needed:**

- Pi-top [4]
- Expansion Plate
- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- "Alex" Robot Model
- Ultrasonic Sensor
- LEDs

**Activity Flow:**

1. Open Further class and the "USB Keyboard" lesson.
2. Implement direction_arrows.py to control the robot via loops and key events.
3. Test controls and optimize performance for the race.

**Reflection Questions:**

1. How responsive was the robot using loops?
2. Did adding more keys slow performance?
3. How did your robot design affect its performance in the race?




# Activity 2: Line Follower (Rover Vision)

**Estimated Time:** 120 minutes

**NGSS Connections:** HS-PS3-3

**TEKS Connections:** P.P.2, P.5(A), P.5(B), P.5(D), P.6(B), P.6(D), P.8(E), A.P.1(B), A.P.1(G), A.2(A), A.12(C), 2A.8(C), G.9(B)

**Activity Overview**

Students use a camera with their Pi-top to implement a line follower program. They learn computer vision concepts such as kernels, centroids, and masks, and integrate Python functions to navigate the robot along a path.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS3-3 | Motion & Vision | Computer vision is instant and always accurate | Students may assume line follower will perfectly track lines without calibration |
| P.P.2, A.P.1(B), A.12(C) | Functions & Programming | Functions are unnecessary for complex tasks | Students may write repetitive code instead of using reusable functions |
| G.9(B) | Sensors & Hardware | Sensors always give perfect data | Students may not account for lighting or contrast affecting line detection |

**Materials Needed:**

- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- “Alex” Robot Model
- Activity Worksheet
- Pencil / Notebook
- Calculator

**Activity Flow:**

1. Show the “Computer Vision” video.
2. Open Further class and complete “Rover Vision” and “Path Finder” lessons.
3. Integrate camera and line detection functions to navigate robot.

4. Test and refine robot performance.

**Reflection Questions:**

1. What is the fastest forward speed achieved?
2. Name two real-life applications of line detection.
3. What potential issues could arise when applying these techniques in real-world situations?
4. What are the benefits of using functions in your code?

## Activity 3: Collision Simulation

**Estimated Time:** 90 minutes

**Lesson Objectives:**
By the end of this activity, students will be able to:

- Understand the relationship between force and energy through experimentation.
- Create a work vs. time graph representing power from collected data.

**NGSS Connections:** HS-PS1, HS-PS3-3

**TEKS Connections:** P.5(A), P.5(D), P.5(E), P.5(F), P.5(G), P.6(B), P.6(D), P.7(A), P.7(D), P.8(E), A.P.1(A), A.P.1(B), A.P.1(C), A.P.1(G), A.2(A), A.12(B), A.12(C), 2A.8(C), G.9(B)

**Materials Needed:**

- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- The “Alex” Robot Model
- Calculator
- Box with a known mass

**Activity Flow:**

1. Navigate to the “Collision Simulation” lesson on Further.
2. Explain Newton’s laws of motion and the relationship between acceleration, force, kinetic energy, work, and power.
3. Students calculate the acceleration of the robot at top speed.
4. Determine the force used by motors and allow robots to collide with an object.
5. Calculate force of the collision, work performed, and plot power vs. time.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS1 | Force & Energy | Mass and force are independent | Students may think a heavier robot has the same effect on the object as a lighter robot |
| HS-PS3-3 | Work & Power | Work is always equal to energy | Students may misinterpret work as just movement without considering force × displacement |
| P.5(D), P.5(E) | Newton's Laws | Newton's laws only apply in idealized conditions | Students may think the laws don't apply to small robots |
| P.6(B), P.6(D) | Measurement & Calculation | Units are not important | Students may calculate force, work, or power without correct units |
| A.P.1(B), A.P.1(C) | Analysis | Graphing is optional | Students may not plot work vs. time accurately, missing trends |

**Reflection Questions:**

1. What happened to the box during the collision? Why did it move?
2. Did the robot maintain the same speed before and after the collision? Why?
3. How would a heavier box or robot affect the results?
4. Which of Newton's Laws of Motion apply in this activity? Explain your reasoning.

---

## Activity 4: Forklift Assembly Challenge

**Estimated Time:** 90 minutes

**Lesson Objectives:**
By the end of this activity, students will be able to:

- Design and build a forklift for their robot to pick up/drop off a wheel-axle assembly.
- Create a program for the robot to approach, pick up, and drop off a wheel-axle assembly.

**NGSS Connections:** HS-PS3–3

**TEKS Connections:** P.5(D), P.5(E), P.5(F), P.5(G), P.6(B), P.6(D), P.7(D), P.8(E), A.P.1(A), A.P.1(B), A.P.1(C), A.P.1(G), A.2(A), A.12(B), A.12(C), 2A.8(C), G.9(B)




**Materials Needed:**

- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- 1 Servo Motor

**Activity Flow:**

1. Navigate to the "Forklift Assembly Challenge" lesson on Further.
2. Discuss train wheel-axle maintenance and consequences of bearing/wheel failure.
3. Show demonstration of an axle being replaced quickly.
4. Students use the Engineering Design Process to design and build a forklift capable of picking up a wheel-axle assembly.
5. Test forklift with angle limits:
    - Maximum lift: 0°– 90°
    - Minimum lift: 0°– -90°

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|
| HS-PS3-3 | Force & Work | All designs function equally | Students may design a forklift without considering leverage or motor power limits |
| P.5(D), P.5(E) | Engineering Design | One design is correct | Students may assume first prototype will succeed without iteration |
| P.6(B), P.6(D) | Measurement | Angles and distances aren't critical | Students may set forklift too high or low, missing constraints |
| A.P.1(A), A.P.1(G) | Problem Solving | Coding logic is optional | Students may not integrate program commands with hardware correctly |
| 2A.8(C) | Testing & Iteration | Test once is sufficient | Students may not evaluate performance or refine their design |

**Reflection Questions:**

1. Did you use your first forklift design? If not, why?
2. Why are angle limits important in your design? What could happen if they are ignored?
3. What aspects of your design were most successful?
4. Did your design take inspiration from real-world applications?




# Day 4: Final Challenge

**Theme:** Applying STEM Skills: Engineering Solutions for Railway Safety

**Day Overview:** Students will apply the Python programming language, hardware programming, and engineering skills they have learned throughout the week to complete a final obstacle course challenge. Teams will collaborate to remotely pilot their robot, swap the locations of two objects, and simulate the real-world process of replacing a wheel-axle assembly. This hands-on activity emphasizes teamwork, coding, sensor integration, and problem-solving in a real-world railway context.

**Lesson Objectives**

By the end of Day 4, students will be able to:

- Modify scripts to control their robot for picking up and dropping off objects.
- Program the robot to drive using servo motors and integrate a camera for remote operation.
- Incorporate sensors such as ultrasonic, light, and line followers to optimize robot performance.
- Apply teamwork, planning, and problem-solving to complete a timed challenge.
- Evaluate hardware and wireless limitations based on their performance during the final task.

## Day 4 at a Glance

| Time | Activity | Focus / Goal |
|---|---|---|
| 15 min | Camp Review | Review key concepts and challenge rules; reinforce prior learning and prepare for the final task |
| 90 min | Forklift Assembly Challenge (Continued) | Design, build, and program a forklift to lift and move a wheel-axle assembly |
| 180–200 min | Final Challenge | Apply the Engineering Design Process to complete the obstacle course, swap axles, and integrate sensors and camera control; practice coding and mechanical design principles |
| 10–15 min | Reflection & Clean-Up | Discuss lessons learned, team collaboration, and real-world applications; disassemble robots and clean up |

## Day 4 Vocabulary

| Term | Student-Friendly Definition | Day 4 Connections |
|---|---|---|
| Scalar | A measurement that only has an amount, not a direction. | On Day 1, students use scalar variables to set baseline values like LED delays, while on Day 4, they analyze numerical motor values to program vehicle operations. |
| Vector | A measurement that includes both an amount and a direction. | On Day 1, students give specific vector-like directions to guide blindfolded teammates, while on Day 4, they program the robot to navigate paths and track coordinates. |
| Speed | How fast an object moves (Speed = Distance / Time). | On Day 2, students write Python algorithms to measure the robot's real-time velocity, which helps them optimize remote steering control during the Day 4 obstacle challenge. |
| Acceleration | How quickly speed or direction changes over time. | On Day 2, students compute changes in motor speeds using internal wheel encoders, preparing them to manage weight shifts while hauling payloads on Day 4. |
| Frequency | The number of wave cycles that pass in one second. | On Day 2, students explore wave signals to program audio alerts, which directly relates to the ultrasonic tracking sweeps used during the Day 4 navigation run. |

## Activity 1: Final Challenge

**Estimated Time:** 180-200 minutes

**NGSS Connections:** HS-PS3-3

**TEKS Connections:** P.5(D), P.5(E), P.5(F), P.5(G), P.6(B), P.6(D), P.7(D), P.8(E), A.P.1(A), A.P.1(B), A.P.1(C), A.P.1(G), A.2(A), A.12(B), A.12(C), 2A.8(C), G.9(B)

**Activity Overview:**
Students review previous lessons, including Python programming concepts, sensor integration, and mechanical operations through discussion and a brief recap of key points. This ensures that teams are prepared to complete the final challenge successfully.

**Misconceptions:**

| Strand / Standard | Key Concept | Common Student Misconceptions | Camp Context Example |
|---|---|---|---|

| | | | |
|---|---|---|---|
| HS-PS3-3 | Forces & Motion | Students may confuse how velocity, acceleration, and force affect the robot | Students might miscalculate the effect of motor force when moving objects |
| P.5(D), P.5(E) | Energy, Work & Power | Students may forget formulas or misapply them during collision simulations | Students may assume mass doesn't affect acceleration or force |
| A.P.1(A), A.P.1(B) | Engineering Design | Students may think programming and sensors can be applied without testing | Students may skip debugging or iterative improvements on the final track |
| A.2(A), A.12(C) | Collaboration & Problem Solving | Students may not effectively communicate within teams | Instructions for axle pickup/drop-off might be unclear due to lack of coordination |

**Materials:**

- Pi-top [4] Robotics Kit
- Laptop with Internet Access
- "Alex" Robot Model
- 2 Servo Motors
- 2 Red LED Lights
- 1 Buzzer
- 1 Ultrasonic Sensor

**Activity Flow:**

1. Open the Further website and access the "Final Challenge" lesson.
2. Review all challenge rules and functionality of robot components (drive, LEDs, buzzer, sensors, camera, servo forklift).
3. Students practice the engineering design process, testing sensor integration, camera control, and movement.
4. Build the obstacle course track and explain the task: pick up the first axle, drive to the end, swap with the second axle, and return to the start.
5. Conduct the timed final challenge, recording completion times and bonus points.

**Reflection Questions:**

1. How much control did you have over the robot? Was modifying the robot speed helpful? Why?
2. Can you add any sensors to help you pick up the axle?
3. Can you add the line follower to help you stay within the track?

4. Was your camera fixed or controlled by a servo motor?
5. What lessons about teamwork, coding, and sensor integration did you learn from this challenge?




# Appendices

**Appendix A: Video Links**

Day 1
Activity 2:
Algorithm Example: Exact Instructions Challenge - THIS is why my kids hate me. | Josh Darnit
What is Python?: What is Python? Why Python is So Popular?
Activity 4:
Traffic Signals: How Do Traffic Signals Work?

Day 2
Activity 1:
Ultrasonic Sensor: How Do Ultrasonic Distance Sensors Work? - The Learning Circuit
Light Sensor (Photoresistor): How does a photoresistor work?
Sound Sensor: Sound Sensors

Activity 3:
The Alex Robot: pi-top [4] Robotics Kit: Build instructions - Alex

Day 3
Activity 2:
Computer Vision: Computer Vision: Crash Course Computer Science #35
Activity 4:
East-Palestine Derailment: 3News Investigates uncovers 6 NTSB probes into Norfolk Southern accidents since 2006
Wheel-Axle Assembly Replacement: Wheel-Axle Assembly Replacement

**Appendix B: Teacher Notes**

| Note # | Teacher Notes |
|---|---|
| 1 | To connect a tablet/iPad or PC to a wireless keyboard using a usb adapter or Bluetooth, first ensure the keyboard is turned on and in pairing mode. On your tablet/iPad or PC, go to Settings, select Bluetooth, and turn it on if necessary. Your device will search for available Bluetooth devices; when the keyboard appears in the list, tap on its name. You may need to enter a pairing code displayed on your screen using the keyboard. Once paired, your device should show the keyboard as connected.<br><br>[Use the pi-top [4] with Wi-Fi and the Further website] (Recommended)<br>Continue after connecting to the pi-top [4] hotspot. Once connected, open a web browser on your Tablet, iPad, or laptop and navigate to the pi-top [4] IP address (192.168.90.1). This should give you access to the pi-top [4] dashboard. Select Wi-Fi Settings and connect to your own Wi-Fi network. After a successful connection, the pi-top [4] will get a different IP address (e.g. 10.0.0.1), check any Further lesson to access and use this new IP address on the Further website.<br><br>[Use the pi-top [4] with the Display Cable and Further/RealVNC Viewer]<br>Alternatively, if you have the green Pi-top [4] display cable, you can use it to connect to your Laptop or Tablet by plugging it into the USB adapter and then connecting the other end to the Pi-top [4] USB-C display port (located next to the charging port). After completing this step, the Pi-top [4] can be accessed through the RealVNC Viewer or the Further website using the IP address 192.168.64.1.<br><br>[Use the pi-top [4] with the Hotspot and RealVNC Viewer]<br>To connect a tablet to the pi-top [4] using Wi-Fi, follow these steps: First, using the pi-top [4] Mini Screen, ensure your pi-top [4] is powered on and has Wi-Fi Hotspot enabled in the Settings page. Now, navigate to the Network page in the pi-top [4] Mini Screen, then look for the hotspot credentials there. On your tablet, go to the Wi-Fi Settings and look for the pi-top [4] network name (SSID) in the list of available networks. Select this network and enter the password. Once connected to the pi-top [4] hotspot, you may open the RealVNC Viewer App and create a connection to the pi-top [4] IP address (192.168.90.1), and connect to that address with the following credentials: Username: *pi*, and Password: *pi-top*. |
| 2 | The Sound Sensor consists of a simple microphone that detects the vibrations of the air entering the sensor and produces an analog reading based on the amplitude of these vibrations. |
| 3 | The Ultrasonic Sensor consists of two main components: a transmitter that sends ultrasonic sound waves forward, and a receiver that detects those waves once they bounce off an object in front (both components are labeled on the sensor as T and R, respectively). The sensor calculates distance based on the time it takes the sound wave to bounce and return to the receiver (distance = speed of sound in the medium * time). |

| | |
|---|---|
| 4 | The Light Sensor consists of a photoresistor and a pair of wires; a photoresistor is a light-sensitive device, the resistance of which decreases (more current can pass through the wire) with the increase in light intensity detected by the sensor. |
| 5 | Instruct your students to remove the Servo Motors and other components that might get damaged at the front of the robot. |
| 6 | Explain to students that a wheel-axle assembly replacement in under 3 minutes is preferred because several train cars may require several wheel-axle assemblies to be replaced while in the railyard. To minimize downtime and avoid costly delays associated with prolonged train stoppages in railyards, optimizing the time needed to complete the wheel-axle assembly replacement is of utmost importance. |

## References

Czernia, D. (n.d.). Car crash calculator. Omni Calculator. https://www.omnicalculator.com/physics/car-crash-force#how-to-calculate-impact-force-g-force-in-car-crashes

Crash Course. (2017, November 15). *Computer Vision: Crash Course Computer Science #35*. YouTube. https://youtu.be/-4E2-0sxVUM?si=mSSksMsn-OD6FgK1

Element14. (2021, January 27). *How Do Ultrasonic Distance Sensors Work?-The Learning Circuit*. YouTube. https://www.youtube.com/watch?v=2ojWO1QNprw

Loveland, T. (2020). *Standards for technological and engineering literacy*. Standards - International Technology and Engineering Educators Association. https://www.iteea.org/stel

MaxBotix. *How Ultrasonic Sensors Work*. MaxBotix Inc. (2023, March 1). https://maxbotix.com/blogs/blog/how-ultrasonic-sensors-work?utm_term=&utm_source=google&utm_medium=cpc&gad_source=1&gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2sVexH654KswgVdUe5BWd3ZmYiHrX-cKI0_7yHonPIav1XD7UbnufsaAkg1EALw_wcB

Next Generation Standards by Topic. (2013). https://www.nextgenscience.org/sites/default/files/HSTopic.pdf

pi-top. (2021, January 22). *pi-top Robotics Kit: Alex Build Instructions*. HubSpot. https://f.hubspotusercontent40.net/hubfs/4368942/Robotics%20Kit%20instructions/pi-top_RoboticsKit_AlexBuild_22012021.pdf

Practical Engineering. (2019, May 14). *How Do Traffic Signals Work?*. YouTube. https://www.youtube.com/watch?v=DP62ogEZgkI




STEM Basics. (2017, February 3). *How Does A Photoresistor Work?*. YouTube. https://www.youtube.com/watch?v=u9Riurh4y9U

STEM Basics. (2020, January 2). *Sound Sensors*. YouTube. https://www.youtube.com/watch?v=H3JacSVipVA

Team, P. (2021, March 9). *Pi-top [4] robotics kit: Build instructions - alex*. YouTube. https://youtu.be/8c-T1KmL0lY?si=RI7HwLM2oSFS8jPJ

Texas Education Agency. (2012). *Math*. https://tea.texas.gov/academics/subject-areas/math

Texas Education Agency. (2021). *Science*. https://tea.texas.gov/academics/subject-areas/science

UTC Railway Safety. (2024, June 20). *Wheel and Axle Replacement.* YouTube. https://youtu.be/bP_T0oEUL78?si=EvKAx_9JjTlt7QSH

*Physics*. StudySmarter UK. (n.d.). https://www.studysmarter.co.uk/explanations/physics/modern-physics/photoresistor/#:~:text=is%20a%20Photoresistor%3F-,A%20photoresistor%2C%20also%20called%20light%2Ddependent%20resistor%20(LDR),light%20and%20increases%20with%20less

**AI Transparency Notice:** Google Gemini 3.8 Flash was utilized during the development of this curriculum strictly as a generative tool to optimize slide layouts, refine visual presentation, and proofread slide text. The underlying educational architecture, data literacy frameworks, and learning activities were authored solely by the project research team.




# Curriculum Writers

Carlos Pena-Caballero

Angela Chapman

Johnathan Sias

Constantine Tarawneh