Seeker®

BY PITSCO EDUCATION

TEACHER'S GUIDE

UNIT 2

Unit Overview

In this Unit, students work with a company called DoorDrop Delivery to deliver food to various locations in a city. The tasks students complete center around using Seeker's drive motors for navigation and its servo motors to move its head and arms. Students learn the benefits of using variables to store information and how algorithms can be used to create step-by-step processes for completing delivery tasks.

In Activity 1, students are introduced to Amara Martin, a logistics coordinator for DoorDrop Delivery. Her goal is to utilize Seeker as a delivery robot to take food orders to specific locations in a city. This real-world robotics application is currently being developed and refined in many urban locations in the USA. To accomplish this, students learn about the importance of algorithms and sequencing when writing programs. They also learn to use variables that store information such as the robot's desired speed, that can be recalled later. Students complete several navigational tasks that teach them how to program Seeker's DC motors to drive forward, backward, make different kinds of turns, and drive in circles and arcs.

Activity 2 centers around Seeker's servo motors. These are the motors that control its head and arms. Servos are positional motors that can rotate between 0 and 180 degrees. Amara wants to expand Seeker's capabilities from delivering food to delivering groceries as well. Students are then introduced to Lee Chen, a supermarket manager, who is struggling with finding employees who can drive and make deliveries. Lee and Amara want to work together to bring groceries to customers who can't make it to the supermarket. Students learn how to program Seeker's arms to pick up bags of groceries and to move Seeker's head so that it looks where it is going when making a turn. They also conduct a servo experiment where they learn about data transformation and graphical representation of data. The activity concludes with the Grocery Delivery Challenge, where Seeker must pick up and deliver bags of groceries to two different locations.

Careers

- Logistics Coordinator
- Supermarket Manager

Context and Setting

A food delivery service in a city using Seeker to deliver food and groceries to customers.

Computer Science and Technology Concepts

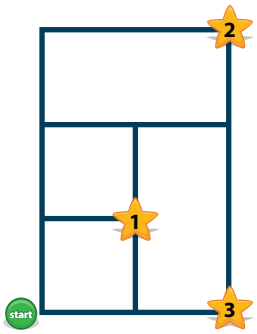
- Algorithms
- Variables
- Data Transformation
- Pseudocode and Flowcharts
- Simulation
- Types of Motors (DC and servo)

Unit Prep Instructions

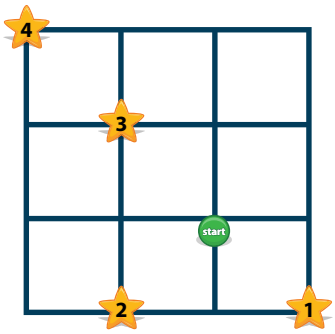
- Before starting this Unit, make sure the Seekers are assembled in robot mode as instructed in Unit 1. Students should not need to do any robot modifications for this Unit.
- Because this Unit focuses on navigation, students will need several courses or maps to test their programs on. Depending on the number of students in your classroom and the space available, you'll need to create areas for testing. All tasks in this Unit can be completed with a 2-meter by 2-meter area. However, there are a couple considerations to think about before creating these testing areas:
    - If students complete the content at their own pace, student pairs will be on different courses at different times. You'll either need different areas for each course, or you'll need to easily be able to take up one course and put another down.

- If you keep the class together as they progress through the content and tasks, it makes it easier to have one testing area with the appropriate course set up for each task.
- You can scale courses to the space you have available. However, doing this will require students to modify their programs from what they are told in the content. In all programming tasks, students are told that a city block is 100 cm.

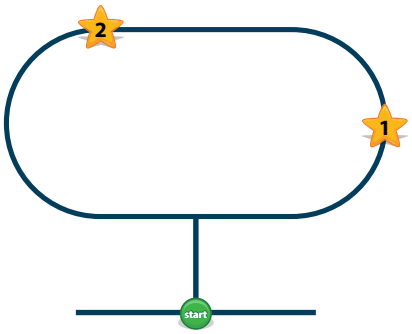
- As you create the course maps for students to test their robot on, consider creating them on poster board that can be taped together. This makes it easy to put down when needed and then picked up and stored when not needed. You can also have one course map on one side and another on the back side.
  - We suggest using painter's tape to make roads for the maps.
  - Here are the course maps that will be required for this Unit:



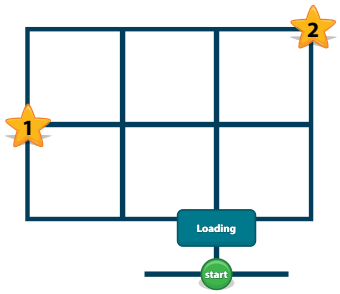
Activity 1, Page 7
Three Deliveries



Activity 1, Page 9
Four Deliveries

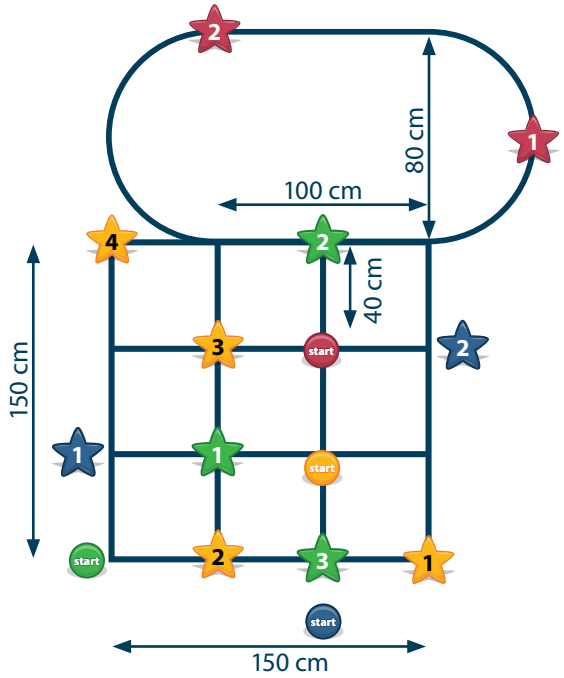


Activity 1, Page 10
Oval Deliveries



Activity 2, Page 12
Challenge

  - To save space and material, the maps can be combined into one city map as shown in the following image. This requires slightly more space because the total length of the map is about 230 cm. But it also makes it possible to run all tasks and challenges on a single map. Consider creating colored stars or other indicators that correspond to each map so students can easily determine where their starting and delivery locations are.



Activity 1, Page 7 – Three Deliveries

Activity 1, Page 9 – Four Deliveries

Activity 1, Page 10 – Oval Deliveries

Activity 2, Page 12 – Challenge

- Consider giving student pairs different roles. For example, for Activity 1, have one student oversee hardware while the other student oversees software. Then, switch roles for Activity 2.

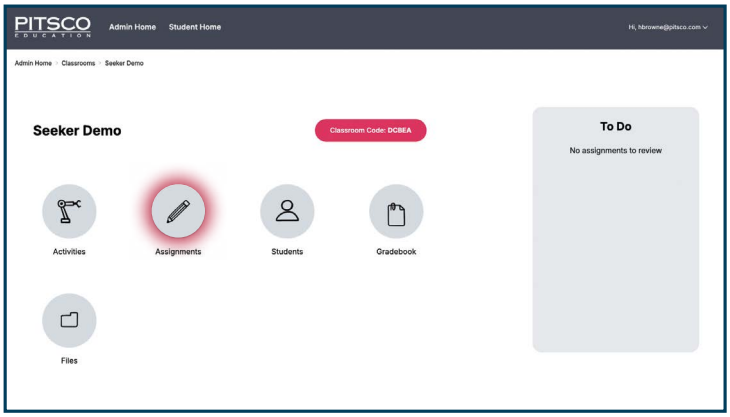
Unit Discussion Questions

- Do you think using robots to make deliveries is practical? Why or why not?
- Besides food and groceries, what other products could robots deliver to customers around town?
- What are some of the positive impacts that using robots to make deliveries would have on society?
- What are some potential negative impacts that using robots to make deliveries could have on society?
- UN Sustainability Goal Number 11 is about sustainable cities and communities. Review this goal's targets and indicators at the following link. Then, discuss the impact delivery robots would have on this goal. **https://sdgs.un.org/goals/goal11**

Unit Assignments

The following assignments are available to be used with Unit 2 content. When you set up your classroom, the points possible default to zero points. If you choose to use an assignment, you will need to set the number of points the assignment is worth. This can be done in the Assignment section when managing your specific classroom.

When a student submits an assignment, it should show up as a To Do item for you to review and assign points.



Activity 1

- Page 4 – Submit Blockly code for driving around the block: U2A1T1-DriveAroundBlock
- Page 5 – Submit updated Blockly code using variables for driving around the block: U2A1T2-Variables
- Page 8 – Submit Blockly code for making three food deliveries: U2A1T3-ThreeDeliveries
- Page 9 – Submit Blockly code for troubleshooting Amara's four food delivery program: U2A1T4-FourDeliveries
- Page 12 – Submit Blockly code for making deliveries on the oval map: U2A1T5-OvalDeliveries

Activity 2

- Page 3 – Submit Blockly code for causing Seeker's head and arms to move: U2A2T1-ServoMovement
- Page 5 – Submit updated Blockly code for using variables as servo positions: U2A2T3-PositionVariables
- Page 8 – Submit Blockly code for completing the basketball play: U2A2T4-BasketballPlay
- Page 12 – Submit Blockly code for completing the Grocery Delivery Challenge

Global Activities

- Career Activity 1: Career Research – Research a career of their choice and submit information about the career in text field.
  - Consider having students research the careers mentioned in this Unit to make a stronger connection between real careers and how robotics is impacting those careers.
    - Logistics Coordinator
    - Supermarket Manager
- Career Activity 2: Career Priorities – In a text field, list priorities for choosing a career, three possible career choices, advantages and disadvantages of each career, and how the career priorities are met with each career.

- Career Activity 3: Interest Profile – Complete an interest profile and record results from the profile in a text field.

Guide to Activity 1: Food Delivery

Page 1 – Amara the Logistics Coordinator

Students meet Amara, who works as a Logistics Coordinator for a company called DoorDrop Delivery. She wants to explore the idea of using robots to deliver food to customers rather than having to rely on delivery drivers who increase the company's carbon footprint. Amara wants to be able to easily program robots to navigate different routes and patterns to make food deliveries.

Tasks to Complete

- Understand the real-world application and context of Unit 2.

Teaching Tips

- Before starting this Unit, make sure Seeker robots are in robot mode. There shouldn't need to be any changes made to how the robots were configured in Unit 1.
- This Unit will involve Seeker driving specific patterns on simulated city maps to make deliveries. To create these maps, you'll need an empty area about 2 meters x 2 meters. See the Unit Prep Instructions for more information on creating the maps.
- Consider having students research the career of a logistics coordinator. Career activities are provided as assignments that can be assigned through the teacher portal.

Notes:

Icon Key

- **Code Saved to Student Profile**
- **Code Testing Task**
- ***Innovation Notebook* Task**
- **Engineering Task**
- **Guided Programming Task**
- **Exploration Task**
- **Sequencing Task**
- **Debugging/ Troubleshooting Task**
- **Challenge Task**

Page 2 – Driving Forward

Lucas guides the students in starting a program to have Seeker drive in a square or around the block. Pseudocode is provided to help guide the students in developing their first navigational program. Lucas explains how blocks represent commands that Seeker completes and how parameters within the block give Seeker specific information for completing the command.

Tasks to Complete

- Using provided pseudocode, begin a program to cause Seeker to drive in a square (drive around the block).
- Investigate the **drive forward block** and its parameters.

Teaching Tips

- Depending on the space you have available in your classroom, you might want students to change the size of the square. The instructions given are for a one-meter square.
- The wait checkbox on **drive forward**, **spin**, and **pivot blocks** can be a little confusing. Make sure students understand the importance of this parameter. When the wait checkbox is checked, Seeker will not move to the next command in the program until the drive parameters for the block are complete. For example, if the command is to drive forward for 20 centimeters, the next command will not execute until Seeker has driven the full 20 centimeters. But, if the wait checkbox is unchecked, the next command will execute immediately. So, if the next command is another block from the Drive category, it will override the previous drive command.

Notes:

Page 3 – Spins and Pivots

Students investigate how Seeker can make turns using **spin** and **pivot blocks**. **Spin blocks** cause Seeker to rotate in place. One wheel rotates forward while the other wheel rotates backward. **Pivot blocks** cause Seeker to turn on one of its wheels. One wheel rotates forward while the pivot wheel remains stationary. Students use **spin blocks** to build a program for driving around the block. Lucas also guides students in how to duplicate blocks that have already been used to complete the program.

Tasks to Complete

- Determine the difference between spin turns and pivot turns.
- Investigate the **spin block** and its parameters.
- Using the provided pseudocode, complete the program for driving in a square (drive around the block).

Teaching Tips

- Consider having students get up from their desks and demonstrate the difference between a spin turn and pivot turn.
- The wait checkbox on **drive forward**, **spin**, and **pivot blocks** can be a little confusing. Make sure students understand the importance of this parameter. When the wait checkbox is checked, Seeker will not move to the next command in the program until the drive parameters for the block are complete. For example, if the command is to drive forward for 20 centimeters, the next command will not execute until Seeker has driven the full 20 centimeters. But, if the wait checkbox is unchecked, the next command will execute immediately. So, if the next command is another block from the Drive category, it will override the previous drive command.

Notes:

Page 4 – Driving Around the Block

Lucas instructs students to save their drive around the block program to their profile. He then discusses how the program is an algorithm, a set of steps for solving a problem or completing a task. Students then test their drive around the block program in the simulator and on their physical robot.

Tasks to Complete

- Save their program to their profile as U2A1T1-DriveAroundBlock.
- Test the program in the simulator and with the real robot.

Teaching Tips

- Students are not given any direction on the method to use for uploading code to Seeker. If they need a refresher on how to upload code, remind them they can always revisit content from Unit 1 to help them remember.
- You might need to tell the students that the city environment in the simulator is scaled to make completing the delivery tasks more realistic. So, even though students are only programming the robot to travel in a 1-meter square, this is scaled up to a full city block in the simulator.

Notes:

Page 5 – Variables

Lucas discusses factors that can affect Seeker's performance in the real-world such as friction and how some parameters might need to be slightly tweaked to get the desired behavior. Students are then guided through the process of changing the speed at which Seeker drives around the block to make the trip faster. After students make all the necessary changes, Lucas discusses how variables can be useful to store data that is referenced multiple times throughout a program. Variables allow programmers to change a value one time, which affects all the places the variable is referenced. Students create and use variables in the algorithm to represent drive speed and the length of a city block. After making these changes to the program, they test it in the simulator and with the real robot.

Tasks to Complete

- Investigate how variables are used to store information in a program.
- Using variables, modify the drive around the block program to drive at full speed.
- Test the program in the simulator and with the real robot.

Teaching Tips

- When naming variables, the programing app will allow any number/letter combination including spaces. However, in most syntax programming languages including Python spaces are not allowed. Encourage students to get in the habit of using a consistent naming convention for variables such as camelCase.
- Consider having a class discussion over these topics:
  - What would happen if the **driveSpeed** variable used in the program were set to zero?
  - What other values could variables be created to represent?

Notes:

Page 6 – Experimentation

Students save their previous program and then do some experimenting to see if they can improve Seeker's accuracy at making food deliveries. Students modify Seeker's drive speed and the length of the city block by changing variables. Students test their changes in the simulator and in the real world. Students answer questions in their *Innovation Notebook* related to the accuracy of Seeker in the simulation and in the real world and what factors might affect Seeker's accuracy.

Tasks to Complete

- Experiment with changing parameters that are stored as variables.
- Use the *Innovation Notebook* to compare the accuracy of the simulator and real Seeker and determine what factors affect Seeker's accuracy.
- Save the updated drive around the block program as U2A1T2-Variables.

Teaching Tips

- If Seeker is not making perfect 90-degree turns, it could be due to a couple factors:
  - The speed at which the wheels turn and whether they slip when they start or finish the turn. This can be helped by making turns at a slower speed.
  - Seeker's turning diameter is determined by its wheel spacing. Over time, the tabs that hold Seeker's motors can flex causing the wheel spacing to be off. Seeker's wheel spacing should be exactly 250 mm from the center of one wheel to the center of the other wheel. If needed, you can adjust the spacing by loosening the set screw that holds the wheel to the motor shaft with a 5/64-inch hex wrench. Slide the wheel in or out on the motor shaft and then retighten the set screw to hold the wheel on the motor shaft.
- See page 14 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 7 – Navigation Exploration

Sophia, Lucas, and Nova (students) split up the duties of programming Seeker to make food deliveries. Students are tasked with exploring other changes to the drive around the block program. They are given a list of objectives to complete which is also provided in their *Innovation Notebook*. Students work on completing as many of the programming objectives as possible in 20 minutes.

Tasks to Complete

- Explore other drive blocks such as **pivot blocks** and the **invert drive direction block**.
- Modify the drive around the block program to complete as many objectives from the *Innovation Notebook* as possible.

Teaching Tips

- This is a self-directed task where students can explore on their own how other drive blocks and drive parameters work. Students are told they have about 20 minutes to explore on their own. Suggested objectives are provided in their *Innovation Notebook*.
- To encourage students to make good use of the 20-minute exploration time:

- Turn it into a small competition to see what group can accomplish the most objectives from the logbook.
- Have a class share time to discuss what each group discovered during their exploration.

- Because explorations are more open-ended, if students want to test their code in the simulator, they might want to use the empty environment. This will allow them to do just about anything without crashing into an object.
- See page 14 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 8 – Puzzle: Delivery Demo

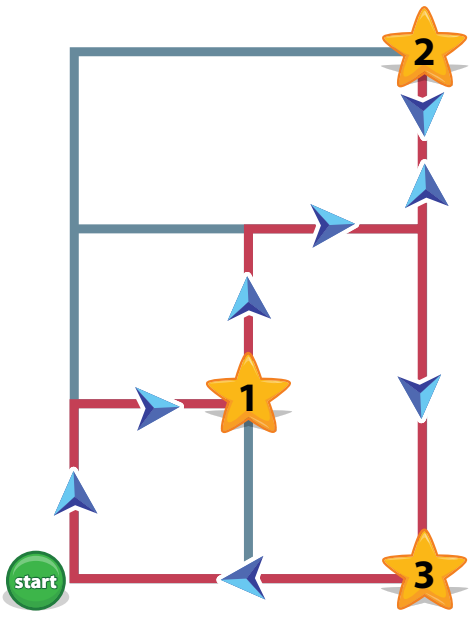
After exploring the **drive blocks** on their own, Lucas and Sophia present the next challenge. Lucas has been working on a program for Seeker to make food deliveries to three different locations. Seeker must start at Happy Sushi and make each delivery in order as shown by the numbered stars on the map. Lucas has already dropped all the needed blocks into the programming area. Students must use the information on the map to connect the blocks in the right order to create the delivery algorithm. After using all the blocks provided, students test their program in the simulation and with a real Seeker on a map created in the classroom.

Tasks to Complete

- Create an algorithm using all the provided blocks to make sushi deliveries in the locations shown on the map.
- Test the delivery algorithm in the simulator and with a real Seeker on the classroom map.
- Save the completed program as U2A1T3-ThreeDeliveries.

Teaching Tips

- To complete this task, students will need to be able to test their code on a map in the classroom. You'll need to create this map in an empty area that is about 2 meters by 2 meters. Consider using painter's tape to mark out the roads for the map. Each square of the map should be 50 cm. This makes the entire map 1.5 meters long and 1 meter wide. Use note cards or other indicators to mark the locations of each delivery.
  - The map can be smaller if needed, but students will need to adjust the length of the city block in their code to match the map you create. Adjusting the length of the block in their code will cause it to not work in the simulator.
  - Consider creating the map on foam board or poster board so it can be taken up and reused multiple times.
  - Students will need to take turns testing their code on the classroom map.



- The important concept with this task is the sequencing of the algorithm over getting Seeker to exactly follow the lines on the map. Students shouldn't spend a lot of time tweaking drive parameters or variables (other than the length of a city block if you've created a different size map).
    - A good rule of thumb is for students to try and keep the lines of the classroom map (roads) between Seeker's drive wheels as it makes deliveries.
- Students will likely need to tweak parameters in their code to get Seeker to complete the deliveries on the classroom map.

Notes:

Page 9 – Troubleshooting Deliveries

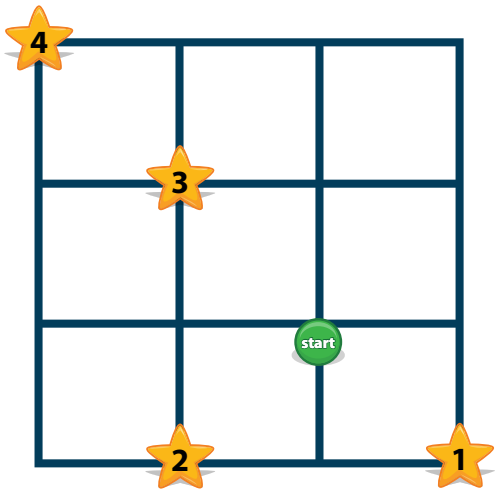
After completing the three deliveries program, Amara tries to up the ante by writing her own four-delivery program. Starting at Dinah's Deli, Seeker needs to drive to each location marked on the map to make a food delivery. However, there are bugs in Amara's program that need to be fixed. Lucas discovers there are three bugs that need to be fixed. Students are tasked with identifying the bugs and fixing them. Students record the bugs they find and their solutions for fixing them in their *Innovation Notebook*. Students use both the simulation and their physical Seeker robot to test their debugged program.

Tasks to Complete

- Debug the provided four-delivery program.
- Record the bugs found in the program and solutions to fix the bugs in the *Innovation Notebook*.
- Test the fixed program in the simulator and with the physical robot on the classroom map.
- Save the working program as U2A1T4-FourDeliveries.

Teaching Tips

- For this task, you'll need to create another classroom map for students to test their code on. The map is a 3 x 3 grid with each square being 50 cm. The easiest way to do this is by adding one more row of squares to the previous map. That should make the map 1.5 x 1.5 meters. Again, use note cards or some other indicator to mark the starting location and the locations for each delivery as shown.
    - The map can be smaller if needed, but students will need to adjust the length of the city block in their code to match the map you create. Adjusting the length of the block in their code will cause it to not work in the simulator.
    - Consider creating the map on foam board or poster board so that it can be taken up and reused multiple times.
    - Students will need to take turns testing their code on the classroom map.



- Students will likely need to tweak parameters in their code to get Seeker to complete the deliveries on the classroom map.

- A good rule of thumb is for students to try and keep the lines of the classroom map (roads) between Seeker's drive wheels as it makes deliveries.

- The important concept for this task is debugging the code. Have students walk through each step of the algorithm to find the bugs. They can test the code as often as needed to help identify the three main bugs in the program.
- See page 14 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 10 – Making Circles and Ovals

After fixing Amara's delivery program, she has one more task for Seeker to demonstrate. She wants to see how Seeker can drive in circles and arcs to make deliveries on curved roads. She presents the Adaptibots team with a challenge of making two deliveries on an oval map. Students begin this challenge by writing pseudocode in their *Innovation Notebook*.

Tasks to Complete

- Write pseudocode in the *Innovation Notebook* for making two food deliveries on an oval map.

Teaching Tips

- Students haven't used the blocks necessary for driving on curved paths yet. But that shouldn't keep them from writing pseudocode to complete this task. Remind them that pseudocode is just instructions written in everyday language.
- Although students will not be using Seeker for this task, it might be helpful for them to see the map the they will use later in the classroom. Recreate the map shown here as best as you can.
  - The distance from the start to the oval should be about 40 cm. The length of the oval straights should be about 100 cm. The diameter of the semicircles should be about 80 cm.
  - The map can be smaller if needed, but students will need to adjust the length of the city block in their code to match the map you create. Adjusting the length of the block in their code will cause it to not work in the simulator.
  - Consider creating the map on foam board or poster board so that it can be taken up and reused multiple times.
- See page 15 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 11 – Circle Predictions

Lucas walks students through how to decompose the oval delivery problem into three parts: the first delivery, the second delivery, and returning to the start at Burger Haven. Lucas then explains how the **start driving block** can be used to control each individual motor to allow Seeker to make turns. He explains that, because there is no distance parameter or wait checkbox on this block, the distance equation (Distance = Speed x Time) needs to be used. Speed is controlled by the motor speeds on the **start driving block**. Time is determined by using a **wait block** after the **start driving block** to tell Seeker how long to drive at those speeds. To better understand the **start driving block**, students are given four **start driving blocks** with different speed parameters for the motors. In their *Innovation Notebook*, they predict Seeker's behavior based on the individual motor speeds. They then test their predictions in the simulator and with their real Seeker robot.

Tasks to Complete

- Investigate how the **start driving block** is used to make circles and arcs.
- Understand the distance equation and how the variables in the equation can be manipulated to affect each other.
- Given four **start driving blocks** with various motor speeds, predict Seeker's behavior in the *Innovation Notebook*.
- Test predictions and record the actual behavior in the *Innovation Notebook*.

Teaching Tips

- The type of navigation students have been using with Seeker is called dead reckoning. It is a method of estimating a robot's position based on its previously known locations, the distance equation, and its heading. It doesn't use any other reference information such as GPS.
- See page 15 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 12 – Oval Deliveries

Students are given a mini challenge of completing the oval deliveries task using only these blocks:

- **Program start block**
- **Program end block**
- **Start driving blocks**
- **Wait blocks**
- **Stop driving blocks**

Students should use their pseudocode and programming knowledge to develop their program that is decomposed into three steps: making the first delivery, making the second delivery, and returning to the start at Burger Haven. Students should test and demonstrate their code working on the physical robot with the classroom map.

Tasks to Complete

- Develop a program for Seeker to complete the oval deliveries mini challenge.
- Demonstrate a working program with Seeker on the classroom map.
- Save the completed program as U2A1T5-OvalDeliveries.

Teaching Tips

- Make sure students are following the criteria and constraints of the challenge by only using the five types of blocks listed above.
- Because the simulator and the classroom map will likely be different, students should build their code to work on the classroom map. This is where their code should be demonstrated.
- It is up to you as to how accurate you want student code to be in order to complete the challenge. Remember, the most important concept is to understand the sequence of events that need to happen in creating their delivery algorithm, not getting Seeker to precisely follow the lines on the map.
    - A good rule of thumb is for students to try and keep the lines of the classroom map (roads) between Seeker's drive wheels as it makes deliveries.
    - Students can spend hours tweaking parameters in code trying to get Seeker to navigate the lines on the map with precision. Sometimes, this can lead to lots of frustration with very little actual learning. Find the right balance of encouraging them to work with grit and determination while also being acceptable of imprecision.

Notes:

Activity 1 *Innovation Notebook* Answer Key

Experimentation

How does the accuracy of the simulation compare to the accuracy of Seeker's movements in the real world?

Answers will vary.

What factors do you think affect Seeker's accuracy in the simulation? What about in the real world?

Answers will vary. Listed here are some example responses.

Simulation: the digital model of the robot, the type of device the simulation runs on, Internet speed, which Internet browser is used, the simulated environment

Real world: friction between wheels and ground, type of terrain, size of the wheels, distance the wheels are apart, Seeker's firmware installed on its controller, the charge of the battery

Navigation Exploration

Check off these exploration objectives as you complete them:

- ❏ In the program, create a variable to represent the 90 degrees used for each turn. Use the variable in each **spin block**. Adjust the variable up or down as needed so Seeker makes a complete square and returns to its original location.
- ❏ Change Seeker's direction to drive in the square in reverse.
- ❏ Program Seeker to make pivot turns rather than spin turns.
- ❏ Change the program so Seeker drives in a triangle rather than a square. It should finish where it started. You'll need to calculate the degrees Seeker needs to turn to make the triangle and change your turning variable.
- ❏ Place an **invert drive direction block** at the beginning of your program and see what it does.
- ❏ Program Seeker to do a 900. That's a 900-degree spin turn. Determine how many rotations that is.

Troubleshooting Deliveries

Answers should be similar to those shown in the following table.

Bug	Explain the bug.	Describe what you did to fix the bug.
1	The turnDegrees variable is never declared or set (initialized) to 90 degrees.	Add a fourth **set variable to block** at the beginning of the program. Use a **number block** to set the **turnDegrees** variable to 90.
2	The program does not complete a movement before moving on to the next command. It jumps from one block to the next without waiting for the command to finish.	Check the wait boxes on all the **spin blocks** and **drive forward blocks**. There are 15 total wait boxes to check.
3	After making the first delivery, Seeker turns to the left instead of to the right to make the second delivery.	After the first **wait block**, change the **spin block** to right instead of left.

Making Circles and Ovals

Write pseudocode where Seeker starts at Burger Haven, makes deliveries at the locations indicated on the map, and returns to Burger Haven.

Pseudocode can vary. The following is example pseudocode for the challenge.

1. Drive forward from Burger Haven to the oval-shaped road.
2. Spin right.
3. Drive forward to the start of the turn.
4. Drive in an arch to the left for a quarter circle to reach delivery location 1.
5. Drive in an arch to the left for another quarter circle to reach the straight road.
6. Drive forward in a straight line to reach delivery location 2.
7. Drive in an arch to the left for a half circle to reach the straight road.
8. Drive forward to the intersection.
9. Turn right.
10. Drive forward to Burger Haven.
11. Stop.

Circle Predictions

Speed Values	Prediction	Actual Behavior
start driving forward left speed % 75 right speed % 75	Answers will vary.	Seeker drives straight forward at 75% speed.
start driving forward left speed % 50 right speed % 75	Answers will vary.	Seeker drives in a gradual arc to the left making a large circle.
start driving forward left speed % 25 right speed % 75	Answers will vary.	Seeker drives in a sharper curve to the left making a smaller circle.
start driving forward left speed % 0 right speed % 75	Answers will vary.	Seeker makes a pivot turn to the left.

Guide to Activity 2: Grocery Delivery

Page 1 – Delivering Groceries

Amara wants to expand Seeker's delivery ability to include delivering groceries. She explains that Seeker will need to be able to lift bags of groceries and deliver them from the grocery store to a customer's house.

Tasks to Complete

- Listen as Amara explains her desire to use Seeker to carry and deliver bags of groceries.

Teaching Tips

- Make sure students have completed Unit 2 Activity 1 before starting this activity.
- This activity will be the first activity where students control servo motors to move Seeker's head and arms.

Notes:

Icon Key

- **Code Saved to Student Profile**
- **Code Testing Task**
- ***Innovation Notebook* Task**
- **Engineering Task**
- **Guided Programming Task**
- **Exploration Task**
- **Sequencing Task**
- **Debugging/ Troubleshooting Task**
- **Challenge Task**

Page 2 – Servo Speed

Lucas explains how servo motors can be used to control Seeker's head and arms so that bags of groceries can be picked up and carried. After explaining how standard servos rotate between 0 and 180 degrees, Lucas guides students in starting a new program to control Seeker's servos. This page focuses on setting the speed of the servos. Students answer questions in their *Innovation Notebook* related to when they would use each block to set multiple servo speeds.

Tasks to Complete

- Understand how servo motors work and what they can be used for.
- Begin a program to control Seeker's head and arms by manipulating servos.
- Answer questions related to setting servo speeds in the *Innovation Notebook*.

Teaching Tips

- If needed, remind students that, when connecting servos to the controller, the black wire goes towards the notch in the servo port.
- Seeker's head should be connected to servo Port 1, Seeker's right arm to Port 2 and Seeker's left arm to Port 4.
- See page 26 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 3 – Servo Position

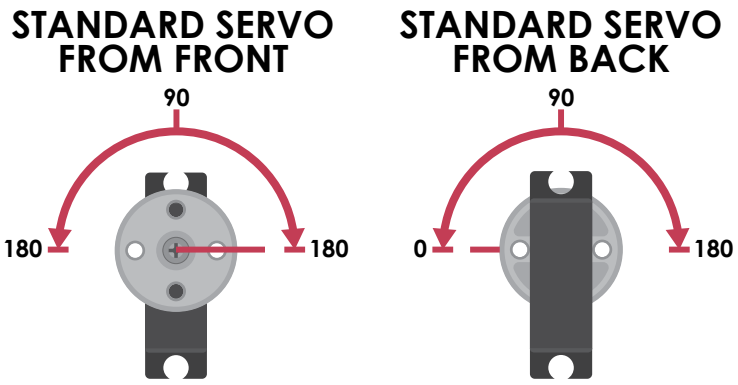
Lucas continues to guide students through creating a program to move Seeker's head and arms. Students learn to use servo set position blocks to move servos to positions between 0 and 180 degrees. After completing the program, students test their program in the simulation and with their Seeker robot.

Tasks to Complete

- Complete a program that causes Seeker's head and arms to move.
- Test the program in the simulator and with the real robot.
- Save their program to their profile as U2A2T1-ServoMovement.

Teaching Tips

- Servo positions can sometimes be tricky to figure out which way is 0 degrees and which way is 180 degrees. If looking at the servo from the back, 0 degrees should be to the left and 180 degrees should be to the right.



- **Servo set position blocks** do not have a wait checkbox to pause the rest of the program while the servos rotate into position. Because of this, it is often necessary to include **wait blocks** after setting a servo's position to give the servos time to rotate into position.
- It is possible for servos to overheat and burn out if too much load is placed on them – for example, trying to lift too heavy of an object. This should not occur for any of the activities developed for Seeker. But, if students are creating their own models, it is important to keep this in mind. If you or students notice that servos are getting warm to the touch or struggling to lift/move a load, turn off Seeker immediately and reduce the load on the servo.

Notes:

Page 4 – Multiple Solutions to a Problem

When students test the previous program, one of Seeker's arms should go up while the other goes down. This is because the servos are opposed to each other. On this page, students are presented with four possible ways to solve the problem. Three of the four programs fix the problem while one does not. Students need to identify the program that does not fix the bug. Clicking **Load Code Example** should load all four programs. Then, students delete three of them to test the remaining program. This process repeats until they find the program that doesn't work. The page concludes with students answering a series of questions in their *Innovation Notebook* about identifying and fixing the bug.

Tasks to Complete

- Load and test four different programs to determine which program doesn't fix the bug of Seeker's arms moving together.
- Compare ways to fix the bug to determine which solution is the best and justify your reasoning.

Teaching Tips

- Discuss with students how there are often multiple ways to solve a problem but that, many times, one solution might be better than others.
- Make sure students use both the simulator and their Seeker robot to find the program that doesn't fix the bug.
- When students click the Load Code Example button, it will load four programs. They need to delete three of the programs and then test the remaining program.
- Consider having students make predictions or take a class poll as to which program won't fix the bug. Can they identify the answer just by comparing the programs and how the blocks are used?
- See page 26 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 5 – Position Variables

Lucas introduces students to the **servo invert block** that reverses the 0- and the 180-degree position of a servo. This allows servos that are opposed to each other to be synchronous using the same position value. Students also learn how to use variables to store positions that can be used in **servo set position blocks**. Using these blocks, students develop a program that causes Seeker to lift grocery bags. They test their program in the simulator and with their Seeker robot.

Tasks to Complete:

- Complete a program that allows Seeker to lift grocery bags.
- Test the program in the simulator and with Seeker.
- Save the program to their profile as U2A2T3-PositionVariables.

Teaching Tips:

- Make sure students understand the value of using variables to store specific servo positions. Using variables allows the positions to be set one time at the beginning of a program. If the value needs to change, it only has to be changed one time rather than every time the servo needs to be moved to that position.
- Remind students to use good programming practices when naming variables. Names should be specific and made up of characters or numbers. Spaces and symbols should be avoided. The names provided in the content use a type of formatting called camelCase.

Notes:

Page 6 – Lee Chen the Supermarket Manager

Students meet Lee Chen, a manager at FarmFresh Supermarket. Lee gives some specifics about his job and why FarmFresh wants to invest in Seeker to make grocery deliveries. Lee challenges the students to put together a demo of Seeker picking up and delivering bags of groceries to customers.

Tasks to Complete

- Understand how robotic technology can be used in the grocery industry.

Teaching Tips

- Grocery and food delivery robots are growing in popularity. As a class, discuss whether these robots are being used in your community or in a community near you. What are or would be the benefits and drawbacks to using delivery robots in your community?

Notes:

Page 7 – Making Grocery Bags

Sophia guides students in making grocery bags that Seeker can pick up and deliver. Students are given the option of using a provided template or engineering their own design. After making three bags of a similar style, students test their bags using Seeker and the U2A2-PositionVariables program. They will likely need to adjust the values for their **armUp** and **armDown** variables to get Seeker to successfully lift and drop the grocery bags.

Tasks to Complete

- Create three paper grocery bags that Seeker can pick up and deliver.
- Test the grocery bags using the U2A2-PositionVariables program.
- Record the values for the **armUp** and **armDown** position variables in the *Innovation Notebook*.

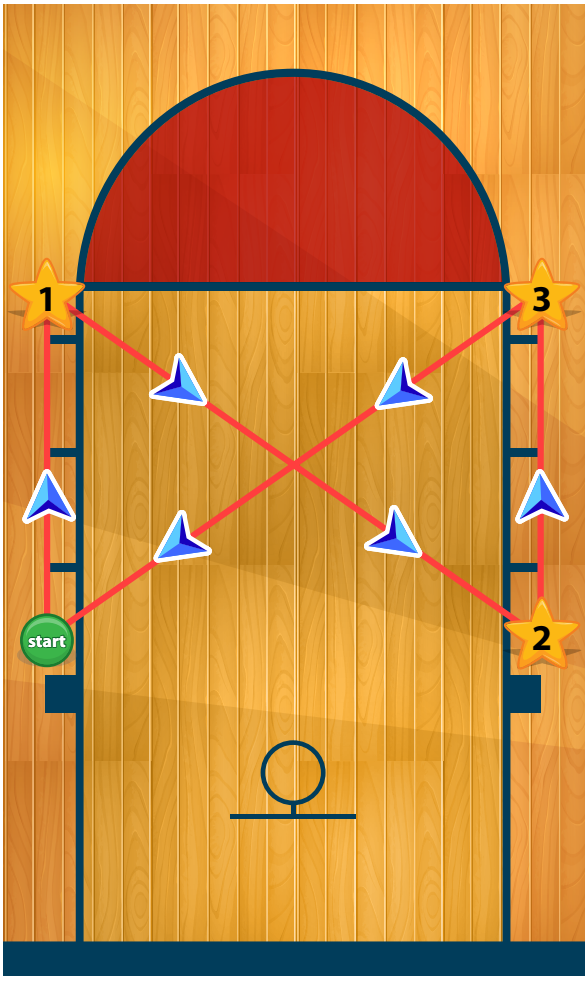
Teaching Tips

- To complete this activity, students will need blank paper, scissors, tape and/or a stapler, and perhaps a ruler.
- The template that is provided can be printed and used to make the creation process quicker and easier. If you want students to use the template, have several copies printed and ready for students to use.
- For more of a challenge, have students engineer their own grocery bags without using the template.
- See page 26 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 8 – Puzzle: Basketball Play

Lucas comes up with a project for having Seeker look in the direction that it turns. To do this, he wants Seeker to drive in a pattern for a basketball play that he's developed as a side project. Lucas has provided all the blocks and the pseudocode for the program. Students just need to put them in the right order. After completing their program, students test the program in the simulator's sandbox and with Seeker to see if it can run the basketball play as described.



Tasks to Complete

- Create an algorithm using all the provided blocks to cause Seeker to drive in the pattern shown for the basketball play.
- Test the algorithm in the simulator and with Seeker on the floor. The robot's head should turn in the direction that it pivots.

Teaching Tips

- When using the simulator, students will need to test their code in the sandbox. Encourage them to turn on Seeker's tracking tail so they can see where Seeker has been.
- It isn't necessary to create a course map for this activity. However, if time permits, have students scale up their distances using the **driveScale** variable in their code and complete the activity on a real basketball court.
- The important concept with this task is the sequencing of the algorithm. Students shouldn't spend a lot of time tweaking drive parameters or variables (other than the **driveScale** variable to fit in the testing area of your classroom).
- There are several cross-curricular geometry concepts that tie into the pattern that Seeker drives for this task. Use the Basketball Geometry Teacher's resource to help explain some of these concepts to your students.
- Have students demonstrate their working code before moving on to the next task.

Notes:

Page 9 – Servo Experiment

Sophia and Lucas discuss the importance of giving servos time to move to their set positions. They guide students in conducting an experiment to measure the time it takes a servo to move from 0 to 180 degrees for various servo speeds. Students record the data they collect in their *Innovation Notebook*.

Tasks to Complete

- Save the previous program to their profile as U2A2T4-BasketballPlay.
- Conduct an experiment to measure the time it takes for a servo to move its full range for various servo speeds and record results in a data table in the *Innovation Notebook*.

Teaching Tips

- This experiment is designed to give students an introduction into data collection and analysis. This concept will be further explored through other Seeker Units.
- Students should be as accurate as possible when measuring the servo movement time. If they start the timer too early or stop it too late, they should run the trial again to get accurate results. If time permits, have them run three tests per servo speed and take an average of the times.
- You can also combine results as a class and find the class average for each servo speed. Discuss how a larger data set should give more accurate results.
- The results from this experiment can be used in other Seeker Units to estimate the amount of time needed for a servo to move from one position to another and a given speed.
- See page 27 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 10 – Visualizing Data

Sophia discusses the benefits of representing data graphically. Students then create a graph of their servo experiment data in their *Innovation Notebook*.

Tasks to Complete

- Create a graph of the servo experiment data.

Teaching Tips

- The shape of the line graph of the student data should follow an exponential decay pattern where the graph starts out with a steep curve and then levels off as it moves to the right. Discuss other real-world examples that follow this same pattern of decay. Some examples are the cooling of a liquid such as hot tea or coffee, the value of an asset such as a car or computer, populations of threatened species, or the amount of drugs (good or bad) in a person's body.
- See page 27 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 11 – Transforming Data

Sophia discusses some of the processes involved in data transformation such as sorting, cleaning, filtering, and looking for trends and relationships. She also discusses how interpolation can be used to estimate data that wasn't directly measured. Students then use their graph and data table from the servo experiment to interpolate the time it would take for the servo to move its full range at speeds not measured.

Tasks to Complete

- Determine the relationship between servo speed and the time it takes the servo to move its full range.
- Interpolate data from measured data.

Teaching Tips

- This page has several data transformation concepts that are touched on. These concepts will be revisited in later Units. But it might be a good idea to discuss these concepts in more detail and with more examples to help students understand them. These concepts include:
  - Data sorting – ordering the data in different ways
  - Data cleaning – looking for outliers or data that seems out of place and trying to determine if the data is legitimate or if an error has occurred
  - Data filtering – removing data that isn't needed or focusing on a smaller subset of the data
  - Interpolation – using known data to predict or estimate an unknown value
- Remind students that the data they gather and the graph they create can be used in later activities to know how much time to give servos to move to their desired positions.
- See page 28 for the *Innovation Notebook* Answer Key for this task.

Notes:

Page 12 – Challenge: Supermarket Delivery

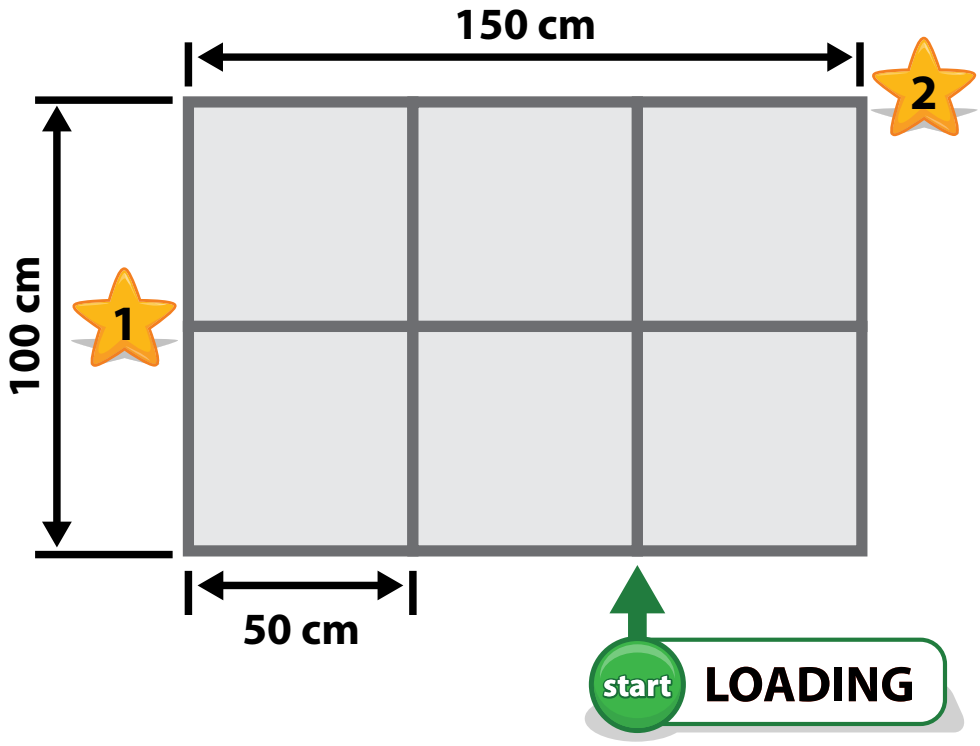
Students are given their first challenge, which is the Grocery Delivery Challenge. The challenge brief is a separate document that outlines the goal of the challenge, the criteria and constraints they must follow, and each objective that must be completed. In short, the challenge requires Seeker be programmed to deliver two bags of groceries to one location, return to the grocery store at the start, deliver another bag to a second location, and then return to the grocery store. To complete the challenge, students must demonstrate their working program on Seeker as well as make a three-minute presentation on their solution and the process for solving the challenge. Portions of the challenge such as writing pseudocode, planning their delivery route, making observations while testing, and outlining their presentation should be completed in the *Innovation Notebook*.

Tasks to Complete

- Create a course for Seeker following the instructions given in the challenge brief.
- Plan the program for making both grocery deliveries.
- Develop the algorithm and program for making both grocery deliveries.
- Test the program in the simulator and with Seeker.
- Demonstrate a working program that successfully makes both grocery deliveries.
- Develop and present a short presentation about the challenge solution.

Teaching Tips

- Decide how much time to give students to complete the challenge. Base your decision on the ability and maturity level of your students, the amount of time you have available for completing this Unit, how precise you want students to be in programming Seeker to make deliveries, and the quality of the presentation that you want them to have. We suggest three to five days. Whatever you decide, make sure students know their deadline.
- Refer to the Grocery Delivery Challenge Brief document for specific information about the challenge. You can find this document in the Teacher Resources section of your classroom in the Learning Portal.
- Students can view the challenge brief document digitally on their device. However, for some students, it might be beneficial to provide printed copies of the document for them to refer to as well.
- The Challenge Brief includes a section on creating a physical test course that resembles a city map. Before starting the challenge, determine if you want each student group to make their own course, whether students groups work together to create a shared course, or whether you create a single classroom course for all groups to use. When making this decision, consider the space available in your classroom and the materials (such as painter's tape) available for creating courses.



- After completing the challenge, discuss the different approaches students had and different routes that Seeker took. How were they similar? How were they different? Discuss how there are usually multiple ways to solve a problem. However, some solutions are usually better than others. What made some solutions better than others?
- As students demonstrate their challenge solution, keep the criteria and constraints of the challenge in mind for assessment and evaluation.
- See page 29 for the *Innovation Notebook* Answer Key for this task.
- The Grocery Delivery Challenge Brief document includes the following grading rubric for assessment.

Category	0 Points	1 Point	2 Points	3 Points	4 Points	Points Earned
Grocery Delivery	Robot successfully delivered zero bags of groceries.	Robot successfully delivered one bag of groceries.	Robot delivered two bags of groceries but with human intervention.	Robot delivered three bags of groceries with a small amount of human intervention.	Robot delivered all three bags of groceries without human intervention.	
Road Navigation	Robot went off a path while navigating to a delivery location four or more times.	Robot went off a path while navigating to a delivery location three times.	Robot went off a path while navigating to a delivery location two times.	Robot went off a path while navigating to a delivery location one time.	Robot did not go off path while navigating to a delivery location.	
Servo Control	A servo (head or arm) was out of position four or more times during the challenge.	A servo (head or arm) was out of position three times during the challenge.	A servo (head or arm) was out of position two times during the challenge.	A servo (head or arm) was out of position one time during the challenge.	Servos were always positioned correctly to pick up and deliver bags of groceries and to turn robot's head to indicate turning direction.	
Program Variables	No variables that had meaningful names were used in the program.	At least one variable that had a meaningful name was used in the program.	At least two variables that had meaningful names were used in the program.	At least three variables that had meaningful names were used in the program.	At least four variables were used in the program and each had a meaningful name.	
Time Limit	The challenge was not completed at all.	N/A	The challenge was not completed within the two-minute time limit.	N/A	The challenge was completed within the two-minute time limit.	
					Total Score:	

Notes:

Page 13 – FarmFresh Partnership

After having completed the Grocery Delivery Challenge, Lee Chen and Amara congratulate the students on a job well done. They discuss the benefits of their companies using Seeker to make deliveries including the positive impacts for people who need food and groceries. The Unit concludes with Amara's company, DoorDrop, partnering with Adaptibots and making multiple Seeker purchases.

Tasks to Complete

- Complete the Activity and the Unit.

Teaching Tips

- If time permits, consider asking some of the discussion questions provided near the start of this document.

Notes:

Activity 2 *Innovation Notebook* Answer Key

Servo Speed

Look at the two blocks for setting the position of one or more servos. What do you think is the difference between these two blocks? When would you use one over the other?

The **servo set position block** lets you set one or all the servos to a single position. You would use this block to synchronize the servo movement together so they all move to the same rotational position. The **servo set positions block** lets you set different positions for each servo at the same time. This block is useful any time you need the servos to rotate at the same time but to different positions.

Multiple Solutions to a Problem

Which program did not fix the bug – top left, top right, bottom left, or bottom right?

The program in the top right does not fix the bug.

What do you think is wrong with this program and how would you fix it?

The **servo invert direction block** is set to false instead of true. When the drop-down menu is set to true, it would invert Servo 4 causing both of Seeker's arm servos (2 and 4) to move together.

Of the three programs that did fix the problem, which do you think is the best solution? Why?

Answers will vary depending on students' opinions. They should be able to justify their reasoning.

Making Grocery Bags

*To lift your grocery bag, what is the value of your **armUp** variable?*

Answers will vary depending on how students created their grocery bags.

*To set the grocery bag back down on the floor, what is the value of your **armDown** variable?*

Answers will vary depending on how students created their grocery bags.

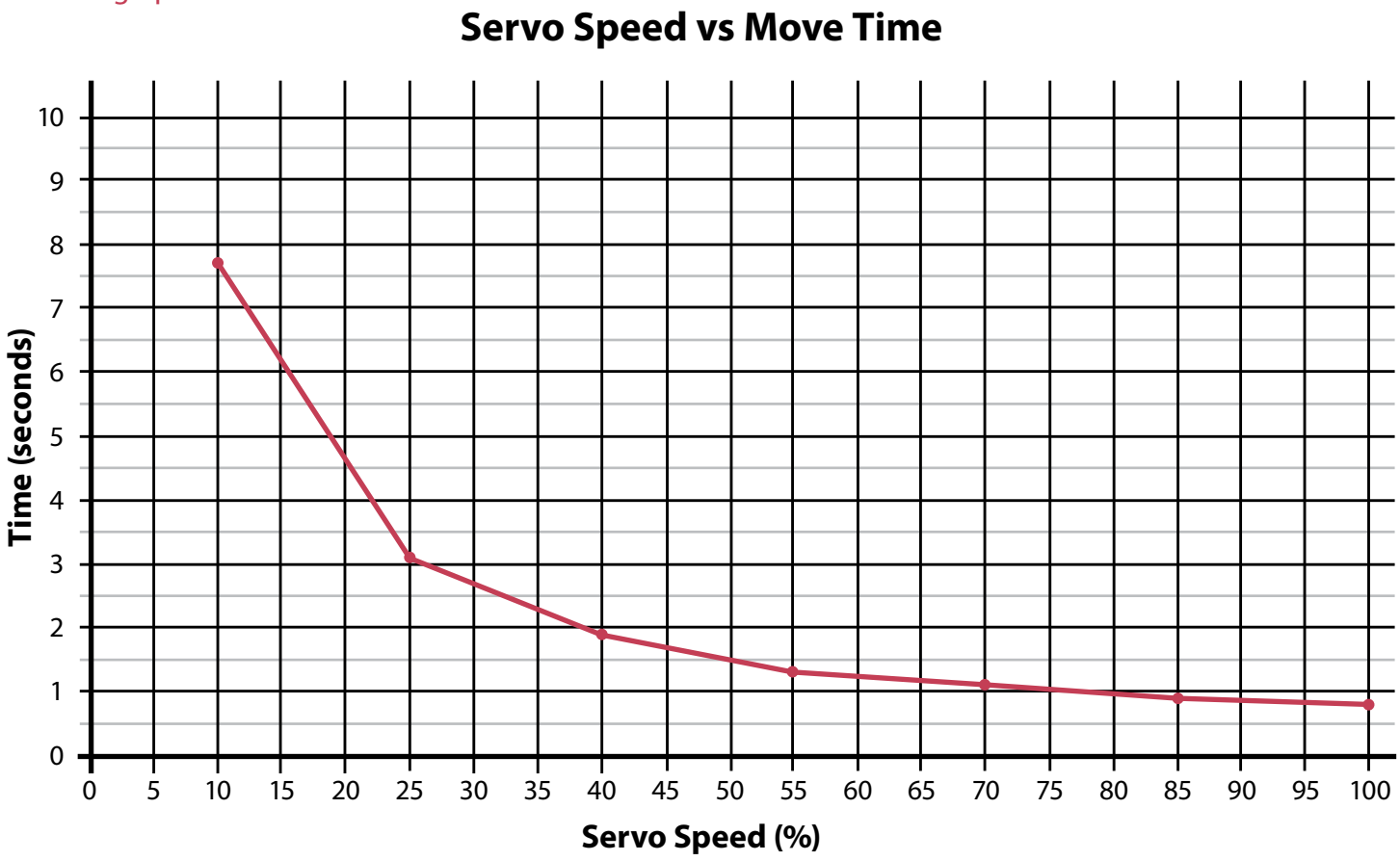
Servo Experiment

Use a stopwatch or timer to determine how much time it takes for Seeker to move a servo from 0 degrees to 180 degrees at the speeds listed in the data table. Record your times in the Time for Movement column.
Times recorded should be similar to those shown here.

Servo Speed (%)	Time for Movement (seconds)
10%	≈7.7 seconds
25%	≈3.1 seconds
40%	≈1.9 seconds
55%	≈1.3 seconds
70%	≈1.1 seconds
85%	≈0.9 seconds
100%	≈0.8 seconds

Visualizing Data

Use the grid to graph your servo speed data. Servo speed is plotted along the horizontal (x) axis. The time it takes the servo to move should be plotted along the vertical (y) axis.
Student graphs should look similar to this.



Transforming Data

Describe the relationship between servo speed percentage and the time it takes for the servo to move.

The higher the servo speed percentage, the less time it takes for the servo to move.

Use the relationship to estimate the time it would take a servo to move the full 180-degree range of motion at a speed of 50%.

Students' answers should depend on the data they collected and their graph. Based on the example graph shown, the estimated time for 50% speed should be about 1.5 seconds.

Challenge: Supermarket Delivery

Plan

Sketch the delivery course. Measure and record distances on your sketch to help with program development. On your sketch, determine what path Seeker will take to travel from the FarmFresh loading area to each delivery location and back.

Sketches will vary depending on the physical course that is created. Encourage students to be as detailed as possible.

Develop pseudocode for Seeker to make the grocery deliveries to each location.

Pseudocode will vary depending on how students go about completing the challenge. It should resemble their final program.

Put to the Test

Program Observation Notes	Program Modification Notes
Observations will vary.	Modifications will vary.

Present

Create an outline of your presentation. You can use your outline as talking points while delivering your presentation.

Presentations outlines will vary depending on the students' solution to the challenge.

Notes:

PITSCO
EDUCATION

Pitsco Education • P.O. Box 1708, Pittsburg, KS 66762 • 800-835-0686 • Pitsco.com
© 2025 Pitsco Education, LLC. All rights reserved.

PE•0225•0000•00 79427