**AP CSP CodeBot**

|  |  |  |
| --- | --- | --- |
| **MISSION 7 Obj 1-5 Hot Pursuit** | | **Time: 45 minutes** |
| **Project Goal:** Students will learn how to calibrate the proximity sensors.  **Learning Targets**   * I can use the proximity sensor to detect objects. * I can experiment with light and dark surfaces to find the ideal power and threshold settings for each environment. * I can write calibration functions so the ‘bot can adapt to its environment. | **Key Concepts**   * CodeBot uses the Infrared Proximity Sensor system to detect objects in its path. * A detection threshold of 0-100% controls how much light is needed for a True detection. If you decrease the thresh value, the ‘bot works well even on a white surface. * An emitter power level setting from 1 to 8 (high power) controls the brightness of CodeBot’s IR “flashlight.” * The prox.detect(power, thresh) function lets you adapt to different environments. | |
| **Assessment Opportunities**   * Mission 7 Obj 1-5 Assignment * Submit the “HotPursuit” program (part 1) | **Success Criteria**   * Use the basic proximity sensor prox.detect() to detect objects in front of the ‘bot * Use prox.range() to find the best threshold for detecting a reflection * Write a calibration function for threshold | |
| **AP CSP Framework**  **DAT-2.E** Explain how programs can be used to gain insight and knowledge from data.  **AAP-2.H** Write conditional statements and determine their results.  **AAP-3.C** Develop procedural abstractions to manage complexity in a program by writing procedures.  **AAP-3.D** Select appropriate libraries or existing code segments to use in creating new programs.  **Computational Practice 2.B** Implement and apply an algorithm.  **Computational Practice 4.C** Identify and correct errors in algorithms and programs, including error discovery through testing. | **Materials**   * A ruler * Different surfaces, like light and dark. * Mission 7 Obj 1-5 Assignment / Answers * Solution code for HotPursuit\_obj5 | |
| **Teacher Notes**   * Objective 1: Students type a short program and then use the proximity LEDs as indicators when they detect an object. They record their results in the assignment. Students will need a ruler and different surfaces for the robot to be placed on. Students put an object, like a folder, in front of the ‘bot and see how close it needs to be before the object is detected. Place the ruler on the surface by the ‘bot for measuring the distance. * Objective 2: Similar to Objective 1, students add a little code and then test the proximity sensors. Record the results in the assignment. * Objective 3: This objective has two parts. First students try the new proximity command in the console panel. Then they add code to the program. The program will print results in the console panel. To slow down the stream of data, students can include a sleep() command. Example code is included in the assignment. * Objective 4: Students will modify the code, but follow the instructions on the assignment to create a function for the threshold calibration. All the code is in CodeTrek, but it should be in a function instead of the main program. * Objective 5: A couple corrections are made in the cal\_thresh() function so that it works without errors. | | |