

## 2. ROVING WITH TOUCH WITH THE PASCO<sub>BOT</sub>

NASA engineers carefully plan a rover's path, but sometimes there are unexpected objects in the way. How can the rover continue to explore if the path is blocked? How can it tell it has encountered an obstacle? What should it do after it finds an obstacle?

### Goals

- Learn how to use the Move with Velocity function block
- Learn about the PASCObot accelerometer.
- Use accelerometer data to help the PASCObot move around obstacles on its own.

### Materials and Equipment

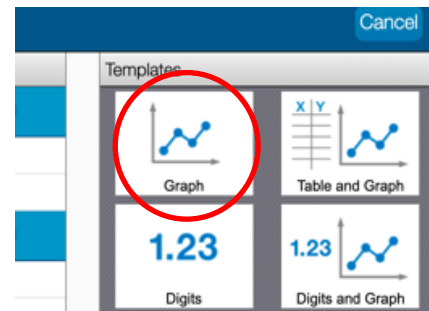
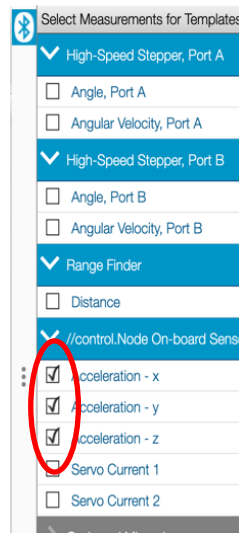
- Computing device
- SPARKvue software
- PASCObot

### Safety

Follow regular laboratory safety precautions.

### Research

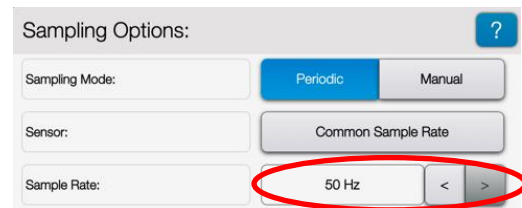
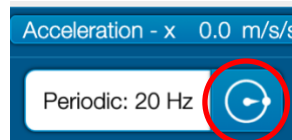
1. If SPARKvue is already running from a previous activity, exit it. Start SPARKvue and connect to the //control.Node.
2. Uncheck each measurement except for **Acceleration-x**, **Acceleration-y**, and **Acceleration-z**, then click the **Graph** template.






Inside the PASCObot is an accelerometer. An accelerometer measures changes in velocity or acceleration. It can measure acceleration in three directions: x, y, and z. Accelerometers are used in many devices including smart phones, video game controllers and rockets. The accelerometers used to deploy airbags have saved many lives. The PASCObot accelerometer will be used to tell when it collides with an object. In the Code window, the **value** block from the **Hardware** menu retrieves the measured acceleration in the x direction. Other measurements and units can be used by clicking on the drop-down menus on the block.







3. Find out which of the three directions (x, y or z) will best measure the acceleration when the PASCObot runs into an object. Because this happens fast, change the data collection rate to 50 Hz.



4. Click on **Start**  and observe the graphs. Hold the PASCObot right side up and carefully shake it forward and back, then side to side, then up and down. Notice which graph (x, y, or z) responds the most to each direction of shake and record them below. Click **Stop**  when done.
5. In the previous activity, the PASCObot was programmed to move a specified distance using the Move a Distance function. In this activity the Move with Velocity function will be used. Open the Code window  and import the **Move with Velocity** function from the PASCO Code Library. Drag the **to moveWithVelocity** block out of the way being careful not to delete it.



6. The **moveWithVelocity** block commands the PASCObot to move at a specified velocity in cm/s. A negative value will cause the PASCObot to move backwards. When the code stops running the PASCObot will stop. That is why there is a 1 s sleep block. Change the velocity to 25 cm/s and place the PASCObot on the floor. Click **Start** . The PASCObot stops moving after 1 s because the program stops running after the sleep block executes.
7. Delete these blocks and drag out a moveWithVelocity block from the **Function** menu. Drag a number block from the **Math** menu and attach it to the **moveWithVelocity** block. Change the velocity to 25 cm/s and click **Start** . What happened?
8. A **repeat** loop from the **Loops** menu is used to keep the PASCObot moving while the program is executing. Selecting **while** and placing a **true** block from the **Logic** menu will cause code within the loop to run until stopped by the user. Place a **repeat while** block from the **Loops** menu around the **moveWithVelocity** block. Place a **true** block from the **Logic** menu in the loop block. Click **Start** . The PASCObot will move until you click **Stop**  or turn it off.

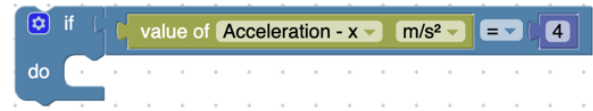


## Prototype

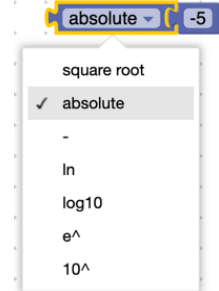
### Program PASCObot to detect and move around obstacles

When the PASCObot runs into an obstacle its velocity changes causing an acceleration mostly in the Acceleration-x direction. This can be used to program the PASCObot to detect objects so it can move around them.

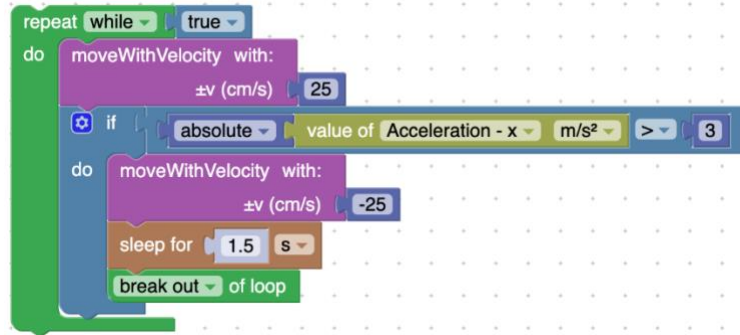
An **if** block from the **Logic** menu will cause code to execute when certain conditions are met. A conditional block from the **Logic** menu can be used with the **if** block to set the conditions. The block shown will execute code placed inside if the acceleration in the x direction equals 4. In addition to the equal sign, the drop-down menu can be used to select less than, greater than, etc. depending on the situation.



Sometimes the sign on a value is not needed. One way to remove the sign is with an **absolute** value block from the **Math** menu. By default, this block is set to **square root**. **Absolute** can be selected from the drop-down menu on the block. The block shown will remove the minus sign and return a value of 5.



1. Create code that will command the PASCObot to go forward until the accelerometer detects it has hit an obstacle. After that, it should back up and stop. An **if** block from the **Logic** menu will help the code decide if an obstacle has been encountered. A **sleep** block from the **Time** menu will give the PASCObot time to back up. An **absolute** value block will cause it to respond whether the acceleration is positive or negative. Sample code is shown below.



## Test

1. Test your code by placing the PASCObot about 0.5 m in front of an obstacle. Click **Start** . Modify the code until it works.
2. Import the **Turn Left** or **Turn Right** functions and the **Move a Distance** function from the PASCO Code Library. Create code that will use the **Turn Left** or **Turn Right** functions and **Move a Distance** function with your existing code to navigate the PASCObot around the obstacle. It should then continue to explore in the original direction. Disable any unused blocks.
3. Place several obstacles around the PASCObot. Modify your code so that the PASCObot will continue to explore after encountering them.

## Improve

1. Sometimes the PASCObot bumps into an obstacle from the side. This can create an acceleration in the y-direction. Modify your code to respond to Acceleration-y and Acceleration-x.
2. Experiment with the obstacle navigation code to reduce the time it takes the PASCObot to go around the obstacles and decrease the chances of it getting stuck.
3. The //control.Node inside the PASCObot has a speaker. Use the **turn //control.Node Speaker on** block in the **Hardware** menu to create a sound when the PASCObot hits an obstacle.

The sample code shown below alternates which way the PASCObot turns when it hits an obstacle. This can help prevent it from getting stuck.

