Unit 5 Robotics Basics

Class Flow

Unit 5 should take approximately 9 hours.

You will need to have RobotC downloaded onto all of the computers ahead of time or plan to spend 30 minutes loading it with the kids. In order to add it to the computers, you will need to create an account with RobotC. If you have the kids do it themselves, they can use either their personal or school e-mail addresses to create the account. I kept the groups the same for this unit as for unit 4 since it was so short and didn't really have a build component. I asked the groups is disassemble their robots before beginning unit 4 and then had them build me a very simple drivetrain for this unit. You could have them keep together their robots from Unit 3 if you'd like but the kids really like to build so it's a nice opportunity to have them disassemble and start fresh.

Lesson 5.1- (Approximately 5 hours)

Objectives

1. Define pseudocode and autonomous programming and its purpose
2. Demonstrate an understanding of coding semantics, syntax, and implementation
3. Identify each part of the code
4. Apply programming best practices for commenting and documentation
5. Introduce variables and conditional statements

First you should spend a few minutes reviewing a good drivetrain. https://www.youtube.com/watch?v=34vB4j89TGs is a video on how to build a good drivetrain. I would show this at this point because their program may be unsuccessful because of a poor build. Their drivetrain should be built using a 1:1 gear reduction or should be geared for torque to help them stay accurate. If the drivetrain is geared for speed, the robots will tend to be less accurate, causing problems.

After you've briefly touched on the drivetrain design, you should go over the slides that walk them through the RobotC environment. If possible, this powerpoint should really be used as a teacher tool and then you walk them through the **actual** environment; they should be clicking while you are clicking and explaining. It took about 1 hour to give the full lecture. At the end of the unit, they should break into their groups and build a very simple drive train then write the code to make their robot accomplish the task: move forward for 2 seconds, stop for 3, and backwards for 5 at half speed. I found that since everyone had the same exact task it was hard to tell if the new code was actually being loaded onto the robot so I would recommend that each member in the group switch up the duration of each task or the speed. So for example one group member would move forward for 5 seconds, stop for 3, and backwards for 2. Or keep the times the same but do it at full speed. It would make it easier to check that everyone has accomplished the task. Most groups were able to build a drivetrain in about an hour and a half and then spent another hour working through the code portion. After all members of the group had the engineering notebook signed off for this task, they were able to move to activities 1-3.

Activities 1-3 took approximately 1 hour each to complete.

For **activity 1**, students should submit the data tables with answers to the questions. They do not need to submit their code as it was provided to them. Possible answers to the questions and guiding questions you should ask your kids along the way about the reason for error:

- With short wait times, the robot doesn't have a chance to get up to full power before the robot stops.
- The robot does not immediately stop at the end of the program but is carrier forward by momentum; particularly true for heavier robots.
- Traction problems at the beginning getting started since you are immediately asking the robot to go to full speed.
- Just because something is true for 500 ms doesn't make it will automatically scale up.
- Low battery juice is not allowing the robot to get to full power every time

For **activity 2**, groups should turn in their code for the assignment. They can answer the two questions at the top of the worksheet as comments in their code or as a separate document. They should also comment the code as far as the behaviors that the robot is doing (ie move forward for 2 seconds at half speed instead of turn left motor on. They should also demonstrate their code for you to sign off on in their engineering notebook.

For **activity 3**, group should turn in the data tables, graphs, and answers to the questions. They do not need to submit code for this. They should see the most error at the short distances with high speed. The short distances with the low speeds tend to be more accurate but sometimes if the robot has high traction wheels or is very heavy, it will not be accurate due to friction.

There really are no "right and wrong" answers here but it's about getting the groups comfortable manipulating the robot's program and showing the relationship between speed and wait states. Some groups may get frustrated that their distances are not working out close to their predictions. The next unit introduces shaft encoders which should help fix most of these problems. One caution is that there will be one natural programmer in the group who will tend to hog the computer. Make sure that you are going around to groups to make sure everyone has a chance to touch the program. You can even assign one person to be the person typing for each activity. Beginning with the next unit, students will be assigned a particular role (programming, project manager, or builder) but for this first one it is important that they all get exposed and get familiar with the programming environment.

Troubleshooting

Coding Lesson 5.1: Activity 0

One wheel turning backwards. Inadvertently checked reverse on left Motor in Motor set up.
Had to update firmware. Motor was plugged into wrong port in cortex.
Right side gears did not move on the backwards command. The last part of the code state 'leftMotor' in each line.
Correction: Change one line of code to 'leftMotor' and the other to 'rightMotor' on the backwards command.
Robot took a long time to move. Correction: Select the USB only command and the robot will run once it is turned on.

Lesson 5.2 - (Approximately 3 hours)

Objectives

1. Explain the difference in turning torque and turning scrub
2. Identify the three adjustments you can make to the drivetrain in order to improve turning accuracy

After all groups have turned in activity 1-3, you should give the powerpoint on turns and drivetrain design. This powerpoint gives details about the way to design a drivetrain to reduce turning scrub and increase turning torque. The main point they should get out of this is that their design is the biggest thing they can change that can affect torque and scrub. The powerpoint should only take about 30 minutes. There is no math in this unit as we aren't going to calculate the actual turning torque or turning scrub. Friction and the effect of friction are not easily defined. The goal is just to get them to understand that they can design out a lot of the problems by designing a good drivetrain. After the powerpoint, you're going to get them to do activity 4 and 5. It took them about an hour to do each activity.

Activity 4 is a turn investigation that took about an hour. They should use trial and error to determine how long it takes to turn 90 degrees. After they have done trial and error for 90 degrees, they should predict how long it takes to go 45 degrees (half as long) and 180 degrees (twice as long). After they have made their predictions, they should actually measure and see how far their robot travels based on the prediction. One of the critiques of this exercise is that it can be difficult to measure the angle in a very precise way. The floor that we used has 12" tiles. Therefore, it was fairly easy to use tape to measure out 45, 90, and 180. However, they weren't able to determine if it went 48 or 50 degrees. I told them not to worry about getting an absolute quantitative value for the actual degrees turned but rather to see if it was exactly right, slightly less/more, or significantly less/more than their predicted values. They should see that as speed increases, the turns get less accurate because of acceleration getting up to speed and momentum after the motors have stopped. They should also see that the point turns were both more reliable and faster. However, some groups found different that the swing turn was more reliable. This could be due to their drivetrain design or other outside factors like wheel style (low traction or high traction). As long as they can defend their answer, there does not have to be an absolute right or wrong.

Activity 5 took about two hours and basically had them put together everything that they learned from activities 1-4. Below is a picture of the course that the students had to maneuver. It can really be any sort of square. I increased the challenge in this activity because activity 1-4 were done on smooth tile, but then the sentry challenge was on a rubber mat. Therefore, some of the calculations that they made in exercises 1-4 needed to be adjusted. However, it gave them more of a chance to see how power level, time, and surface affected the robot. They at least had a starting point for their time and power level and should have chosen to use something around half speed since that had some of the best accuracy. Too slow and the robot runs into too much friction when trying to turn on the rubber mats that we used and too fast and it loses too much precision. Some of the kids got frustrated because there were a number of variables that impacted their ability to stay on the mats such as low battery power, friction/traction problems, and slightly uneven floors. This is part of the exercise. They should get frustrated because using time to measure distance is not the best way. They will learn in the next lesson a more reliable method.

In the end, every group was able to successfully accomplish the task. Some groups finished quickly while others took a while longer. One way to help keep the fast finishing teams engaged is to offer a few bonus points for: the group that goes through the sentry challenge the fastest (faster is harder since you lose accuracy), the team who can get their robot to go through it twice without adjustment, or the team who can have their robot go through the course forward and then backwards without adjustments. I also did not make the kids do both a point turn and a swing turn for the sake of time.

For the answers to the questions on the worksheet, if they make their wheels smaller, they should see that their robot will not travel as far since the RPM is staying constant but now their wheel circumference is smaller. As the battery power gets more and more drained, the robot will not travel as far. Some groups may find that point turns were more reliable than swing turns and vise versa. Some of this has to do with the robot design. Generally speaking, the swing turns were more reliable for most of my groups.

Quiz - (approximately 1 hour)

I created a kahoot as a quiz review. This covers most of the multiple choice questions that appear on the quiz. In addition, I went over commenting some code and talked about proper syntax. The quiz review lasted approximately 30 minutes and then the students took about 30 minutes to complete the quiz. The biggest part that kids missed on the quiz for the commenting portion was that they thought you had to have a delay at the end after the last turn motors off piece of code. However, the delay is not necessary. The error was that the closing bracket was missing. It was a good opportunity to reiterate why the delay wasn't necessary (it's at the very end of the code and your goal is just to turn the motors off, which is accomplished with just the two lines of code that tell the motors to turn off). There is a bank of questions available so that you can mix it up each year.

https://play.kahoot.it/#/k/eb4242e5-3876-46f9-9d75-20de7341da98

General Information:

Troubleshooting tips:

- Make sure the motors are plugged into the ports that are declared in the code. The original assignment says ports 2 and 3 so I would recommend just sticking with this. It's also built into the rubric.
- You may have a case where the motors are spinning in opposite directions; to fix this you can either use programming to reverse one of the motors in the motors and sensors set up or you can physically flip it where the motor controller connects with the motor.
- Make sure the motors are actually plugged into the motor ports and not the analog or digital ports.
- Make sure your communication mode is USB. It will work if it is VEXnet or USB but it waits 10 seconds to check for a joystick before going to USB.
- You also want to make sure you are in VEX 2.0 environment, not VEX IQ. If your motors and sensors tab says something about VEX IQ then you know you need to change your mode.