Servos

July 10, 2018

Introduction

This guide will cover how to use a REV Smart Servo. Servos are useful for small mechanisms and can function as continuous or angular. In continuous configuration, servos function like motors and rotate based on a set power. In angular configuration, servos can be commanded to go to certain positions with a maximum limit of 280°.

Table of Contents

Servo Programmer

The REV SRS Programmer sets the mode of the servo to either continuous or angular. It can also set the limits for an angular servo. The SRS Programmer requires 4 AA batteries.

A REV Smart Servo can still be used without the programmer. By default, the servo is in angular mode. If you want to change the servo to continuous mode or change the limits of the servo, you have to use the servo programmer.

There are 4 buttons and 2 switches on the servo programmer (See picture below). The servo plugs in with the white wire on the prong labeled with an S. The 2 black wires plug into + and -.

OFF-ON: This switch turns the servo programmer off and on. Upon being turned on, all 4 LEDs will flash. If the servo is left on for too long, the LEDs will flash as a reminder to turn the programmer off.

C-S: This switch sets the programmer between continuous and angular modes. C is for continuous, and S is for angular.

Program: This button is generally used for programming the servo. Setting the servo programmer to continuous (C-S switch to C) and pressing program will put the servo into continuous mode. The servo will be set to angular if the C-S switch is set to S. After a successful program, the program LED will flash briefly then stay illuminated.

Continuous

As stated before, the servo can be set to continuous mode by switching the servo programmer to C and pressing program. The program LED should flash briefly and stay solid. Once the servo is in continuous mode, it can be tested using the test button. Pressing test once will cause the servo to rotate back and forth. Pressing the test button twice will initiate manual testing. Pressing left causes the servo to rotate counterclockwise, pressing right causes it to rotate clockwise. Pressing test a third time causes the servo to stop.

On the software side of continuous servos, the servo must be declared as a continuous rotation servo in the hardware map. Writing code for a continuous rotation servo is identical to motors. servoName.setPower(1); causes the servo to rotate one direction, and -1 causes the other direction. In Blocks, set power is under Actuators > CRServo.

Angular

The servo can be set to angular mode by switching the servo programmer to S and pressing program. Pressing test once will cause the servo to cycle between its left and right limits. Pressing test twice will enter manual testing, and the servo will go to its 0° position. Pressing left while in manual testing mode causes the servo to go to its left limit, and pressing right causes the servo to move to its right limit. Pressing test a third time causes the servo to stop.

If you want to change the servo's angular limits, I would recommend setting the servo to its 0° position first. Then, manually rotate the servo to the desired left limit and press the left button. The left LED will blink then remain solid. Rotate the servo to the desired right limit and press the right button. The right LED will blink then remain solid. Once both the left and right LEDs are solid, press the program button. The program LED will blink then remain solid. Now that the servo limits are configured, you can test them using the testing steps written in the previous paragraph.

Generally for angular servos you will use servoName.setPosition(double);. The range of setPosition() is between 0 and 1; 0 being the left limit, and 1 being the right limit. In Blocks, set position is under Actuators > Servo.

servoName.getPosition(); gets the position of the servo, however, it only gets the position that the servo is supposed to be at, not the position it is currently at. For example, if you set the servo's position to 1, but something is preventing the servo from moving, servoName.getPosition() will return 1 even though the servo is not at 1. I would recommend using a potentiometer if you want the true position of your servo, but at that point you may as well use a motor.

Contact Information

Feel free to ask me anything about this guide or any other questions you have about robotics! Send emails to engineering@yale.edu or you can message me on Discord at nahn20#5234. I use Github to upload the example teleops in these guides. All Yale School of Engineering & Applied Science guides can be found here.