Team 3176 White Paper

Title: Resetting Swerve Encoders	Authors: Sam Cordry, Kyle Phillips, Stuart Goedde
Subteam: Programming	Subject: Swerve Pod Encoders
Season: 2018/19	White Paper Number 2

Table of Contents

Abstract

This paper documents the process and proof for resetting the encoders of a robots. The paper is oriented for the general FIRST community interested in learning the basics of resetting encoders for movement controls. The goal of resetting encoders is to allow the encoders to direct the same way so the robot can moves correctly.

Definitions

Encoder- A device that sends the measure of the rotation of the motor that it is connected to, to the RoboRIO.

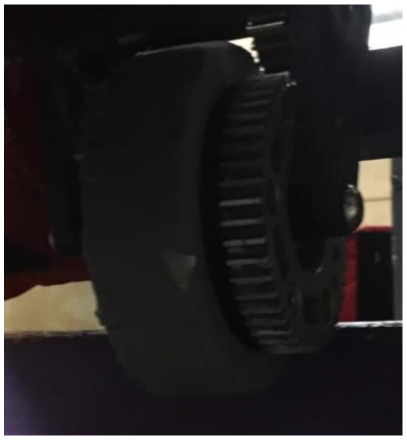
Process

In this white paper, we will be using an application called Phoenix Tuner to receive the current absolute values of the encoders.

If you try to make your robot go in a certain direction, but any of the wheels face a different direction, you will need to reset the encoders. Follow the steps below to reset them.

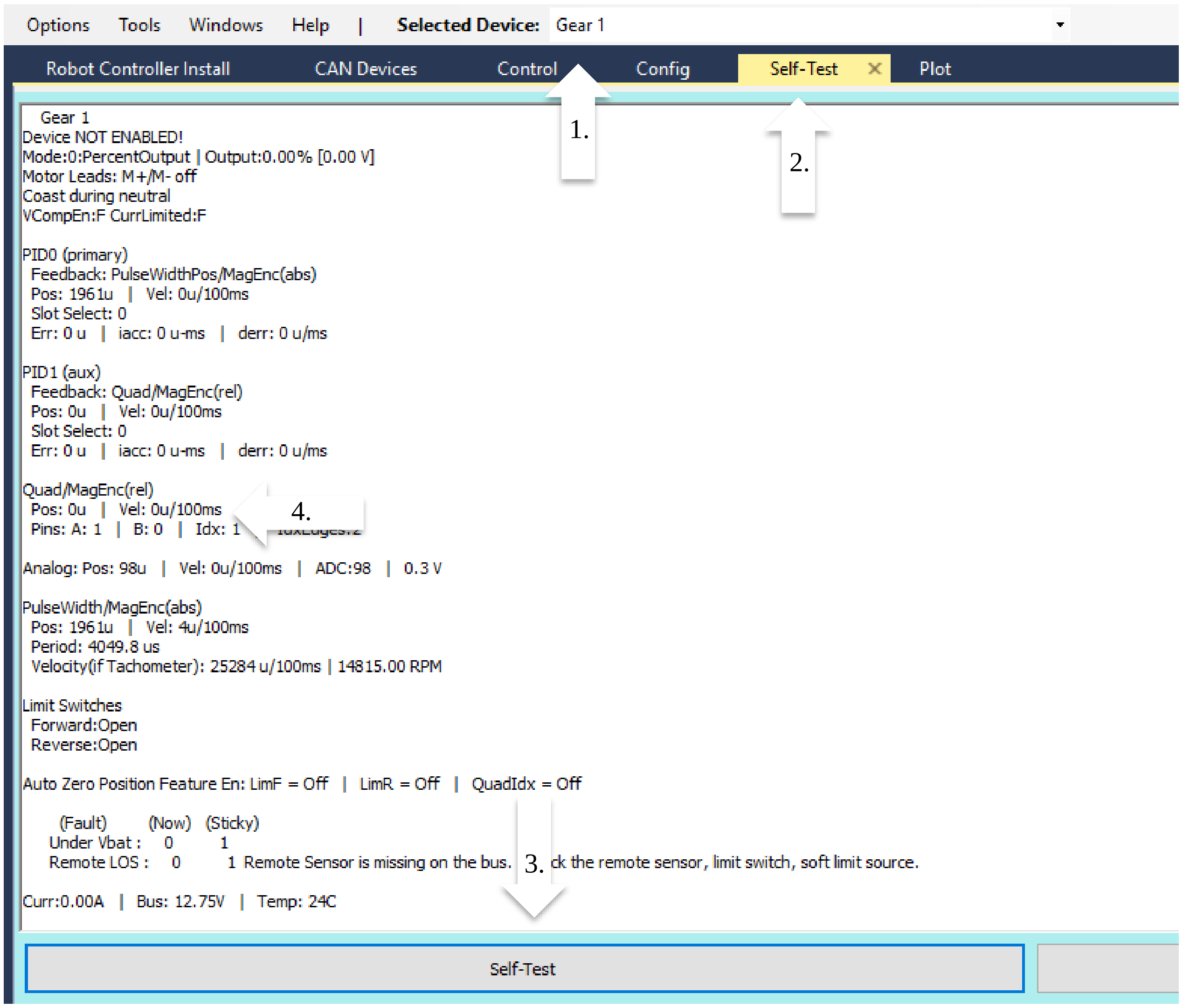
Steps

1. First, you need to manually set the wheels to face the same direction. We recommend orienting the wheels to be facing the front, as this is an easy direction to identify.



All wheels should be oriented this way.

2. Once you have done that, connect the computer (with Phoenix Tuner) to the RoboRIO via a printer cable.
3. Open the Phoenix Tuner application, select a swerve pod from the dropdown menu and navigate over to the self-test tab. It will show up as “Gear (number)”, make sure you know which gear is which encoder.
4. In Phoenix Tuner, add the absolute offset of each pod to an offset variable for each swerve pod. This offset should be taken into account in your code when setting the encoder values. It is essential that you use the absolute value, not the relative value as these give you drastically different values.



```
  Gear 1
Device NOT ENABLED!
Mode:0:PercentOutput | Output:0.00% [0.00 V]
Motor Leads: M+/M- off
Coast during neutral
VCompEn:F CurrLimited:F

PID0 (primary)
 Feedback: PulseWidthPos/MagEnc(abs)
 Pos: 1961u  |  Vel: 0u/100ms
 Slot Select: 0
 Err: 0 u  |  iacc: 0 u-ms  |  derr: 0 u/ms

PID1 (aux)
 Feedback: Quad/MagEnc(rel)
 Pos: 0u  |  Vel: 0u/100ms
 Slot Select: 0
 Err: 0 u  |  iacc: 0 u-ms  |  derr: 0 u/ms

Quad/MagEnc(rel)
 Pos: 0u  |  Vel: 0u/100ms
 Pins: A: 1  |  B: 0  |  Idx: 1

Analog: Pos: 98u  |  Vel: 0u/100ms  |  ADC:98  |  0.3 V

PulseWidth/MagEnc(abs)
 Pos: 1961u  |  Vel: 4u/100ms
 Period: 4049.8 us
 Velocity(if Tachometer): 25284 u/100ms | 14815.00 RPM

Limit Switches
 Forward:Open
 Reverse:Open

Auto Zero Position Feature En: LimF = Off  |  LimR = Off  |  QuadIdx = Off

     (Fault)    (Now)  (Sticky)
   Under Vbat :   0       1
   Remote LOS :   0       1  Remote Sensor is missing on the bus.  ck the remote sensor, limit switch, soft limit source.

Curr:0.00A  |  Bus: 12.75V  |  Temp: 24C
```

Key:
1. Specify encoder with this dropdown menu.
2. Click the “Self-Test” tab.
3. Click the “Self-Test” button.
4. Use the “Pos:” value as your offset value for the specific encoder.

Offsets

In our team, we made a class called "Constants.java" that stores all the variables that will never change throughout the execution of the program. We define an array that contains the offsets for the encoders.

```
public static double OFFSETS[] = {2991.0,2201.0,3746.0,3392.0};
```

Once changed, it will automatically be taken into account in our code.

```
private double findSteerPosition(double wantedAngle) {
        encoderPosition = steerMotor.getSelectedSensorPosition(0) - kConstants[id];
        radianPosition = encoderUnitsToRadian(encoderPosition);
        radianError = wantedAngle - radianPosition;
        if(Math.abs(radianError) > (3*PI/2)) {
                radianError -= Math.copySign(2*PI, radianError);
        }
        else if (Math.abs(radianError) > (PI/2)) {
                radianError -= Math.copySign(PI, radianError);
                velocitySetpoint = -velocitySetpoint;
        }
        encoderError = radianToEncoderUnits(radianError);
        driveCommand = encoderError + encoderPosition + kConstants[id];
        return (driveCommand);
}
```