Robotics Organization

Plan

1. Pre-Season: Teaching new programmers and relearning ourselves
   1. Way around the robot and software (what do all the parts of the robots mean, how to turn it on/off)
      1. Go over document
      2. Give a visual tour of the robot
      3. Look at software being used and how it works in conjunction with the robot
   2. Code foundaions of a robot upward (getting a better understanding of what the code is doing)
      1. Big steps:
         1. Reprogramming Waffle
         2. Starting a robot from scratch
         3. Tuning up Archie
      2. Small steps:
         1. Getting a light blinking
         2. Setting up a WPILIB Project
            1. Make a subsystem (part of the robot) and command
            2. Building and deploying code
         3. Learn about control loops
   3. 2024 repo (GitHub repository, “giant filing cabinet”)
2. Build Season

A robot consists of

1. Physical elements
   1. Output (physical movement)
      1. Motors (move robot and parts)
         1. Continuous motion (set velocity or acceleration)
         2. Servo (set angle)
      2. Motor controllers (control the voltage passed to the motors) [Ex: Can Spark Max, CTRE Talon, CTRE Falcon]
      3. Pistons (move pneumatic parts of the robot)
      4. Solenoids (control pistons)
   2. Robot input (uncontrollable)
      1. Camera (see from the robot’s perspective; detects shapes, colors, april tags, etc.)
      2. LimeLight (camera that detects and tracks reflectivity)
      3. Encoders (measure revolutions, the speed the motors turn)
      4. Gyros (measure of acceleration: rate of change of velocity [rate of change of position], orientation)
      5. Sensors (distance sensors, lidar [Light Detection and Ranging], etc.)
      6. Limit switch (a button that reads from the motor controller, can stop a robot from overextending)
   3. Human input (controllable)
      1. Buttons (boolean trigger [0 or 1]) [Ex: A, B, X, Y, joystick buttons, bumpers]
      2. Trigger (1 axis value [double from 0 to 1])
      3. Joystick (2 axis values, x and y)
      4. D-pad/POV (angle [integer from 0 to 8, times 45])
2. Code
   1. Output (giving info to physical elements)
      1. Physical change
         1. Pass a value to the motor controllers (motors [double])
         2. Changing a solenoid state (open/close a piston [boolean])
         3. PID controller (passes a value to the motor controllers [distance from a target, where you want to go]) [smart with the movement of the robot, corrects for error in the output]
      2. Digital change
         1. Changing a camera setting (LimeLight)
         2. Updating values in SmartDashboard
         3. Turn lights on/off
         4. Calibration of sensors
   2. Input (uncontrollable, receiving info from physical elements)
      1. Reading sensor values
      2. Reading controller values
   3. Processing
      1. Kinematics (move the arm up 1 m, forward 1 m; calculate 2 angles; give them to a PID controller)
      2. Image processing (retroreflective tape; LimeLight’s input; trigonometry for distance; movement to PID controller)
      3. Sequencing actions (move arm down; grab object; move arm up and forward; drop object)
      4. Motion profiling (positions moved-to based on subpositions or using PathWeaver)

What makes up a WPILIB project in Visual Studio Code:

1. Subsystems (access part of the robot)
2. Commands (tell subsystems what to do with parts of the robot)
3. Gradle (build system, collects all written code and sends it to the robot)

Getting a robot set up with code:

1. Flashing the Rio to make sure it has firmware (give them an operating system)
2. Set up the radio (a wifi access point, allows you to connect from the computer to the robot without a USB cord)
3. Make sure motor controllers and sensors that you’re using are connected (CANBUS, a 2-wire cord connection like USB)
4. Deploy code to the robot

Code:

1. Software that connects physical controllers to physical/digital output (mapping human input to robot output)
2. Drive train (how the robot moves around the field and orients)
3. Camera input
4. Parts of the robot