PROGRAMMING

Robotics takes hardware and software, without one the other is useless. Our team uses the object-oriented programming language Java, along with Android Studio, to program the robot. We spent the early part of the season getting our software up to date and ready. We started the season off strong, making sure we were prepared as we progressed through the year.

On the programming team, problem-solving is an important task we use to improve and master the next obstacle. Our programmers overcome obstacles by never giving up even when the wall seems to be too high. Programming is more than just software. Programming is about discovering problems and working together to find efficient ways to solve them.

Autonomous

Our team continues to improve autonomous every chance we get. If it fails we try again, if it works we make it better. First, our autonomous program was made to use a camera and read the non-customized cone. The base code was found off a FIRST site, then a bit of tweaking was done to the code and we were all set. We had our first autonomous program down for the season. The code worked very consistently for zone one and two, however zone three was harder to detect. We decided to tell the robot that if the camera does not indicate zone one or two, then it must be zone three. With our camera-based autonomous program, it read the cone fairly accurately, but was still inconsistent at times. We did not want to settle, so we began looking for solutions to this problem.

```
runtime.reset();
while (runtime.time() < 2){
    double green = colorSensor.green();
    double blue = colorSensor.blue();
    double red = colorSensor.red();
    totalGreenValue = totalGreenValue + green - 50;
    totalBlueValue = totalBlueValue + blue;
    totalRedValue = totalRedValue + red;
    count = count + 1;
}
```

*records color values over two seconds

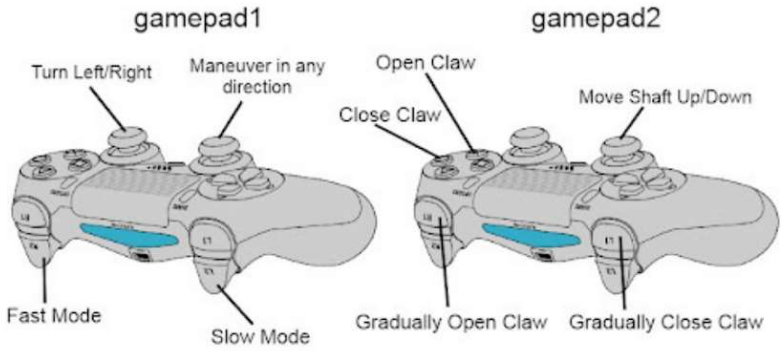
```
if (avgBlue > avgRed && avgBlue > avgGreen){
    turn( degrees: 7);
    zone1();
    zone = "1";
}

else if (avgRed > avgBlue && avgRed > avgGreen){
    turn( degrees: 7);
    zone2();
    zone = "2";
}

else if (avgGreen > avgRed && avgGreen > avgBlue){
    turn( degrees: 7);
    zone3();
    zone = "3";
}
```

*records color values over two seconds

Eventually, we decided to ditch the camera entirely, and use a color sensor instead. This new program reads our custom cone for a total of two seconds, and finds the average red, green, and blue value over those two seconds. After looking at these values it makes a decision on where to go based on what value was highest.

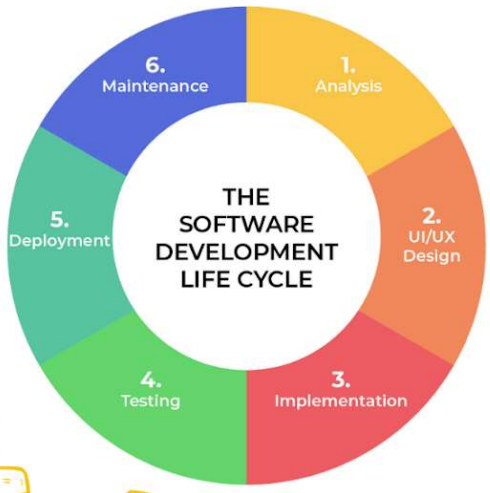
Tele-op

Tele-op is simpler than autonomous. Our main tele-op program is field centric. This means that no matter where our robot is facing, pushing right on the joystick, for example, will move the robot to the right side of the field. This is accomplished using our macanum wheels along with trigonometry and vectors to keep track of our robots orientation. As for the shaft and claw, we use encoders to keep track of their position so that they don't go too high or too low and risk breaking the mechanism.



The Software Development Life Cycle

When we program, we make sure to follow the Software Development Life Cycle, a process of planning to make programming both simpler and more professional. Anytime we program, we are always working on a stage of this life cycle. Things often don't go the way we want, that's just something you have to accept when programming; but, by using this process we are able to better keep track of what we are doing, and what we need to do. Additionally, the process allows us programmers to remain better connected to what we are doing.



Resources

- Learn Java For FTC by Alan Smith
- Tensor Flow Lite: ftc-ml.firstinspires.org/
- A LOT of Stack Overflow and Reddit
- Odometry 101 For FIRST Tech Challenge Robots by BrBatanga on YouTube