**Autonomous objectives:**
During the autonomous period, the robot does the following (sensors are bolded):

1. Inspects and remembers our custom Signal's AprilTag, using its Logitech **webcam** and available **EasyOpenCV** with an **AprilTags plugin**
2. Navigates to the medium junction in front of it, relying heavily on **gyroscopic heading correction**, **odometry**, custom **LIDAR** "nose" and its **whisker** (**vibration sensor**)
3. Delivers its starting cone
4. Navigates towards its cone pile with its **webcam** using a **custom OpenCV pipeline**
5. Uses its AutoGrab **LIDAR** and Paws to pick up a new cone. The robot's paws have **limit switches** to prevent penalties from moving the cone
6. Racing to the high junction with another cone in hand (or, in paw) while using **heading correction**, as well as, of course **LIDAR** to adjust strafing distance to deliver the cone while on the run when the **whisker** detects the junction.
7. A second round of picking up another cone and delivering it to the high junction
8. Finally parks in the correct position based on the AprilTag read in Step #1

**Sensors used:**

- **"Whisker" - Vibration Sensor:** Automatically drop on Junction while moving
- **'Nerves" - Paw limit switches:** Keep paws gentle, particularly when picking up from cone stack
- **"Nose" - LIDAR** (Distance sensor on servo): Detecting cones during auto-grab as well as distance correction to have correct range to drop on Junctions.
- **"Eyes" - Logitech Webcam:** Signal cone inspection, autonomous navigation to cones
- **Encoders on Drive Train**: Distance tracking and odometry
- **Gyroscope/IMU**: Field-oriented driving as well as Turning and Heading Correction
- **Linear Slide Touch sensor:** Set lower bounds on linear slide, protecting string as well as simplifying driving
- **Linear Slide Encoder:** Correctly lifting grabber to right place, as well as preventing lifting slide too much
- **Color Sensor:** Vestigial Signal Cone inspector, replaced by WebCam

**Key algorithms:**
Despite the fact that only one of us had robot-programming experience before this season, or maybe because of this, it was very important to us that we understand the math behind our robot's code. This meant programming several functions almost entirely from scratch: the holonomic drive-train, field-oriented driving, incorporating several different sensors, and automated Assist routine features into Teleop, especially useful forgrabbing and dropping of cones.

Of particular interest is how we programmed the drive train.

- We started with a conversation on conceptually how the angled wheels can move in any of the 4 primary directions — forward, back, right, and left — which we called North, South, East, and West.

- We wrote down the four Clockwise (+1) or Counterclockwise (-1) directions for each wheel to achieve North, South, East, and West. We knew that opposite wheels probably always did the same thing, but we thought direct documentation was the best way to learn
- With these settings, we programmed, tested (and fixed!) movement in the four directions
- For all the angles in between these four (eg, with Joystick driving), we found the two primary directions that surrounded the requested angle and blended amounts of those primary directions depending on how close each was to the requested angle. For example, due NE would be equal amounts of North and East.
- This proportional combination of two primary directions gave us the full-speed power to go in the desired direction, and we then multiplied it by the desired power fraction.

**Driver controlled enhancements:**

Our AutoGrab feature is triggered with GamePad 2's right trigger and does the following while the robot continues to be driven towards a cone:

- Raises our Spike out of the way
- Opens the paws
- Turns on the LIDAR with a close (10cm) limit. This scans the area between the paws.
- Waits for a cone to be detected
- When a cone is detected in range, the paws are brought together and both limit switches are engaged
- The Spike is lowered, grabs the cone and lifts it

This has greatly decreased the time to grab a cone.

**Engineering portfolio references:**

Software programming section
Whisker Section