Control Award Submission Form

****Please turn in this sheet during your judge interview along with your engineering portfolio****

Team # 8696	**Team Name: Trobotix**

Autonomous objectives:
- Read the team prop's position and place the purple pixel on the correct spike mark.
- Navigate to score a yellow pixel in the correct randomized position on the backdrop.
- Park backstage.

Sensors used:

Logitech Webcam	Streams images to our custom pipeline that detects prop placement.
Odometry Pods x3	For accurate robot localization during autonomous.
Slide Encoder (built-in)	Prevents the linear slide from exceeding it's maximum length or unspooling.

Key algorithms:
- Gaussian blur combined with HSV color masks and contour detection to locate objects in images.
- Three dead wheel localization algorithm that accounts for heading / turning drift using a process known as the "pose exponential".
- PIDF controllers, one per driving, strafing, and turning.
- Finite-state machine for pixel detection.

Driver controlled enhancements:
- Toggleable drive between field-centric and robot-centric.
- Improved controller sensitivity by squaring the raw value, attaining a linear rate-of-change.
- Toggleable reduced speed mode, for when finer drive movements are desired.
- Toggleable "turn everything on" mode, where the active intake and internal transfer system are turned on and angled to pick up pixels.
- Automatic internal transfer system stop when carrying two pixels.

Driver interfaces:
- Driver configurable autonomous. Intuitive telemetry-based UI for configuring autonomous parameters. Drivers can, for example, calibrate the HSV color values used in the autonomous image processing pipeline to account for lighting variations. See next page for sample.
- Autonomous scripting. A custom scripting language for programming autonomous routes. More intuitive and has a simpler syntax compared to Java.

Example:

```
GoToShortOrLongRoute
ShortRoute
   GoToTotemSection
   SectionTotemCenterRed
      drive          800      -8.72     178.76
      intake         up
      itsPower       -1.0
      wait_seconds 0.5
      itsPower       0.0
      drive          543.91 -4.69      178.76
...
```

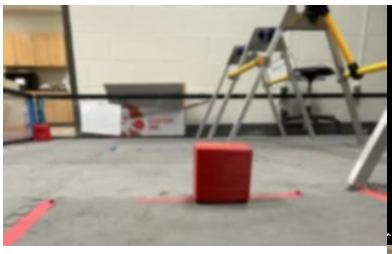
Engineering portfolio references:

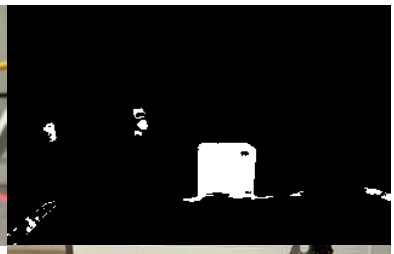
Design § 3.7.b	Sensors
Programming § 4.1	TeleOp
Programming § 4.2	Autonomous

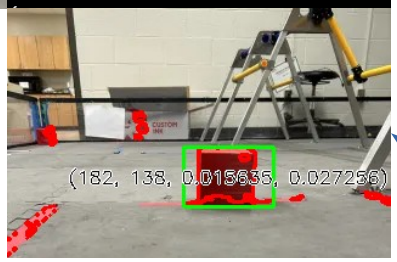
Additional information:

Image processing steps:

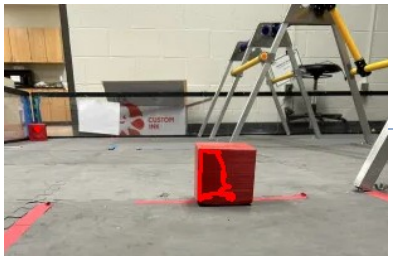
- Apply a gaussian blur to reduce noise.
- Convert the color space from RGB to HSV.
  - Difficult to filter colors in RGB.
  - HSV resistance to lighting changes.
- Mask image to filter for certain colors.
- Detect contours (e.g. edges) based on the mask.
- Find the largest object.
  - Calculate the “pixel totality” per contour, the number of colored pixels in an object divided by the total pixels in the image.
  - Ignore objects that do not meet the minimum totality.
- Assume the largest object is the prop. Locate spike mark.
  - If no prop is detected, assume it’s on the rightmost spike mark.

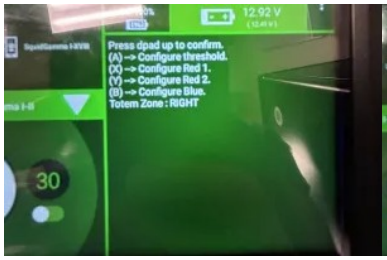


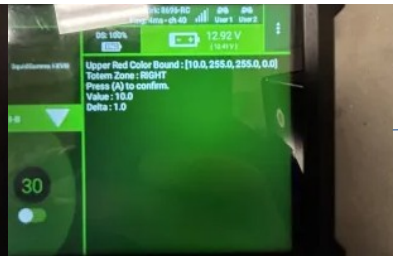




Sample image processing calibration:







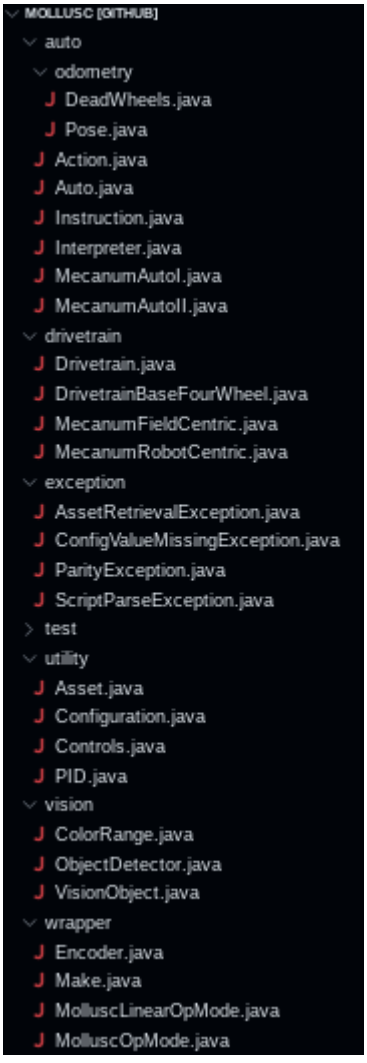
Software design:

- This year, we developed our own library, Mollusc, to improve organization.
- By categorizing common functionality into reusable modules, we can more effectively adapt our codebase to meet new functionality.
- An intimate understanding of the library also reduces confusion.

Odometry debug emulator:

- Created a emulator to test and debug issues in our localization calculations.

Mollusc source tree:



Autonomous program sample diagram: