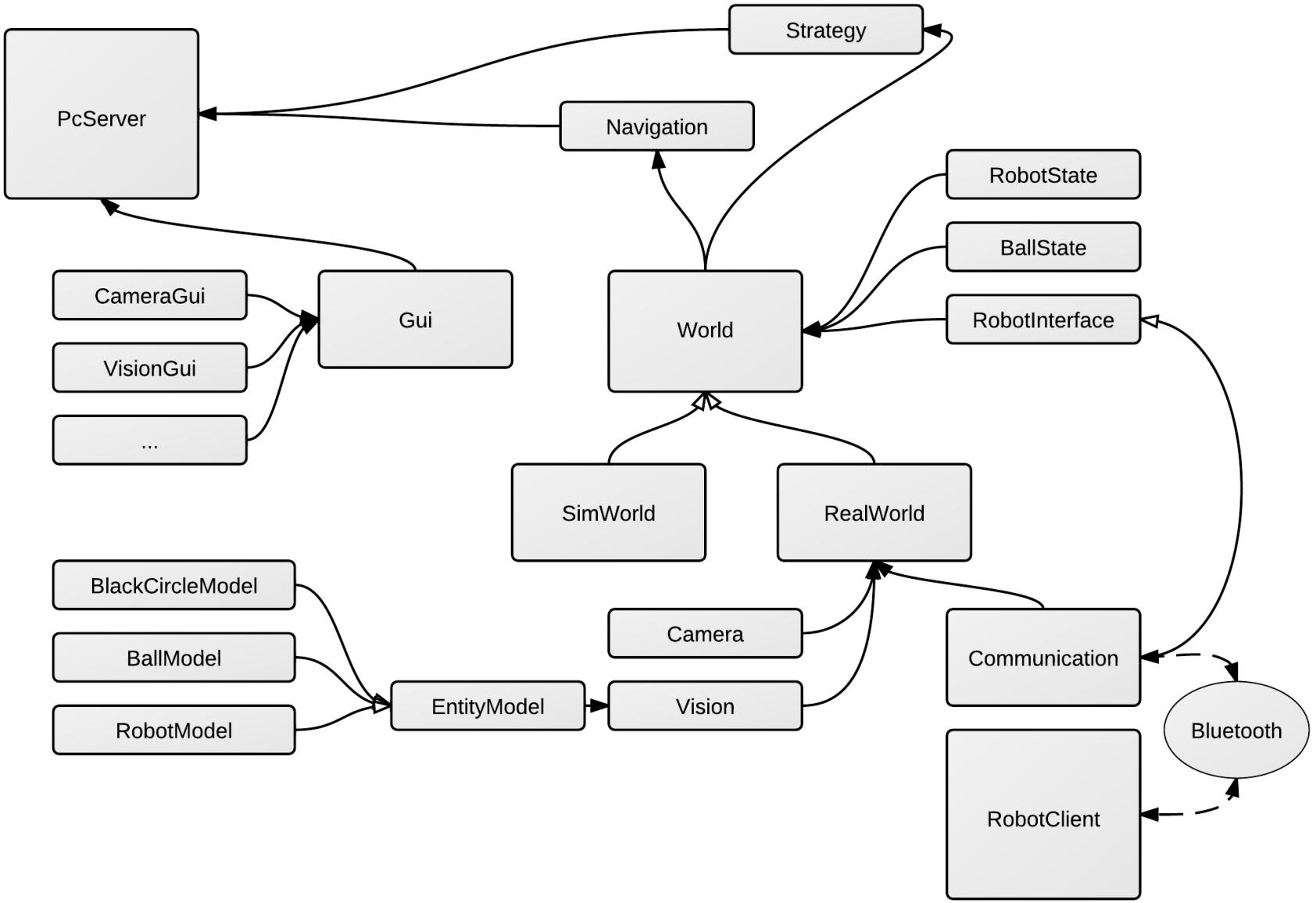
Software

Software!

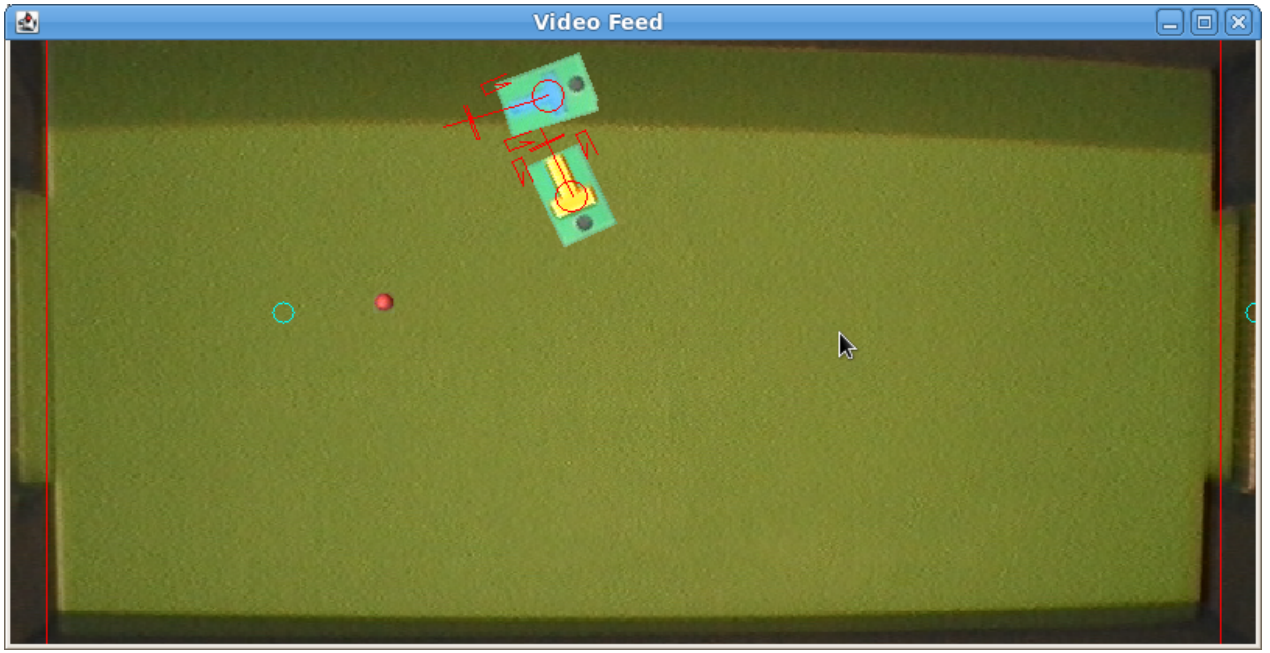
First Steps: Design



First Steps: Communication

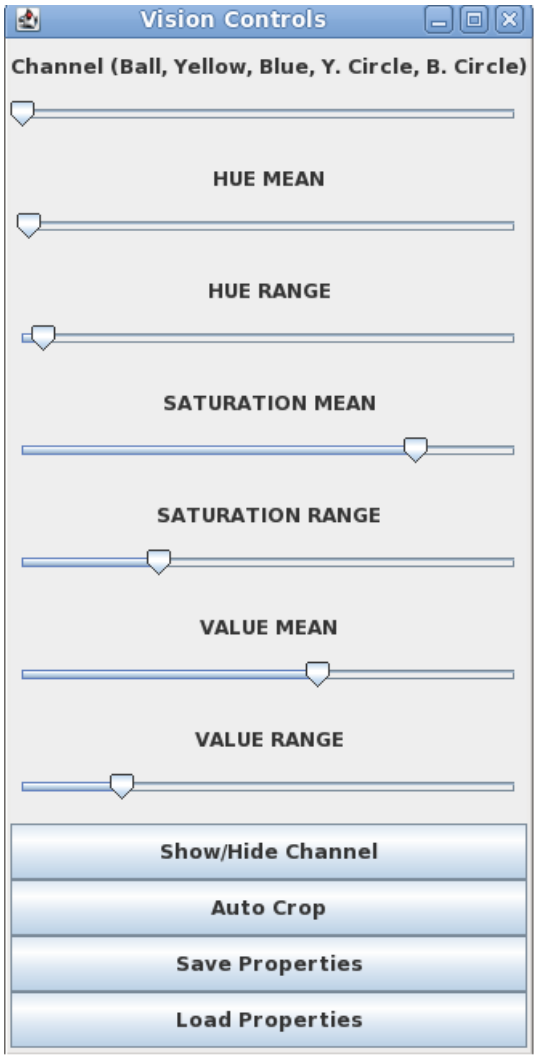
- Two packet types: update wheel speeds & kick.
- Wheel speeds updated every frame.
- About ~100 ms PC-to-robot latency.

First Steps: Simulator



- Simple physics
- One keyboard controlled robot
- One AI controlled robot

First Steps: GUI



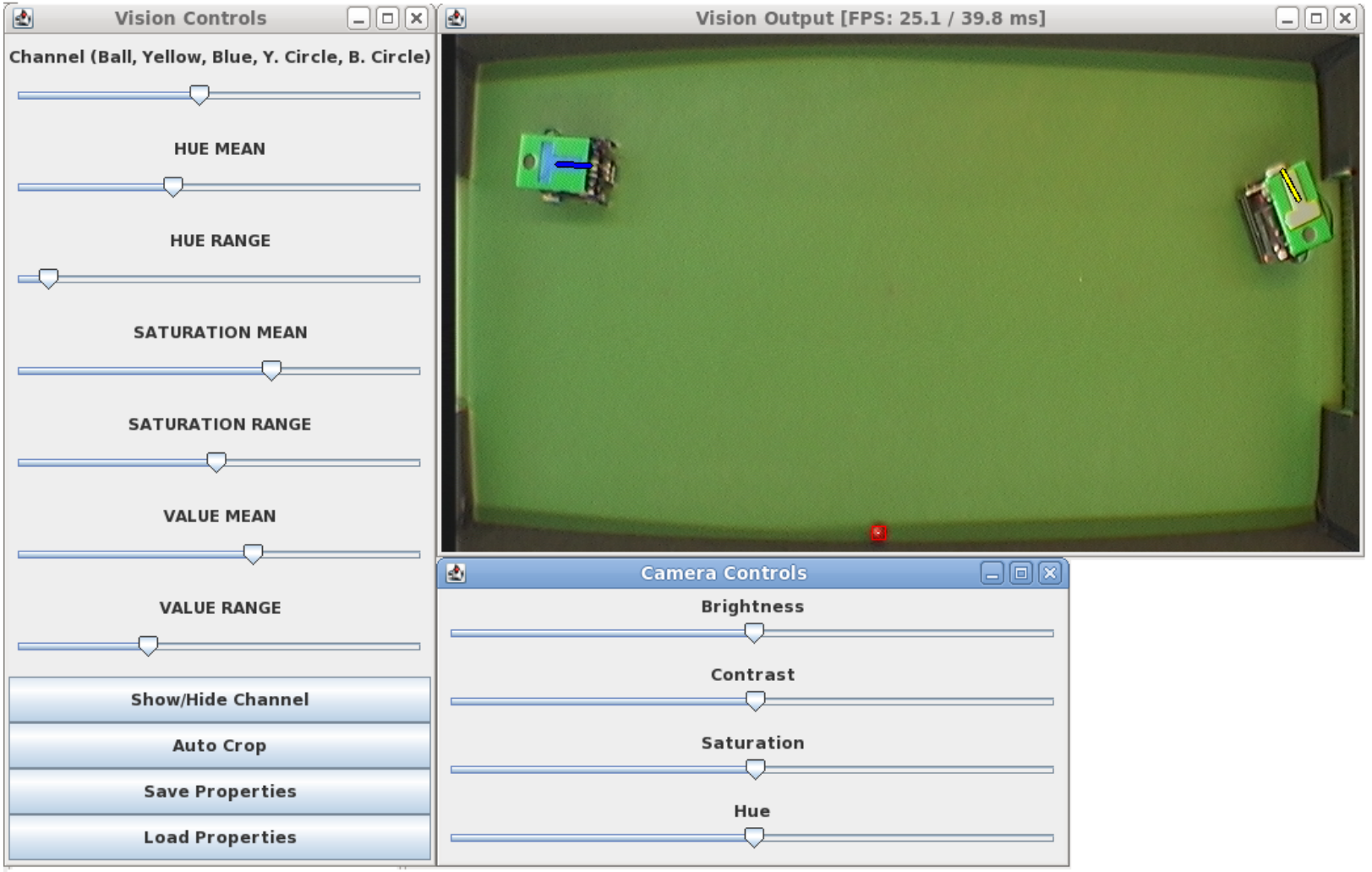
- Easy to add sliders to tweak parameters.
- Easy to save/load parameters from files.
- Essential for rapid prototyping.

Time Compression

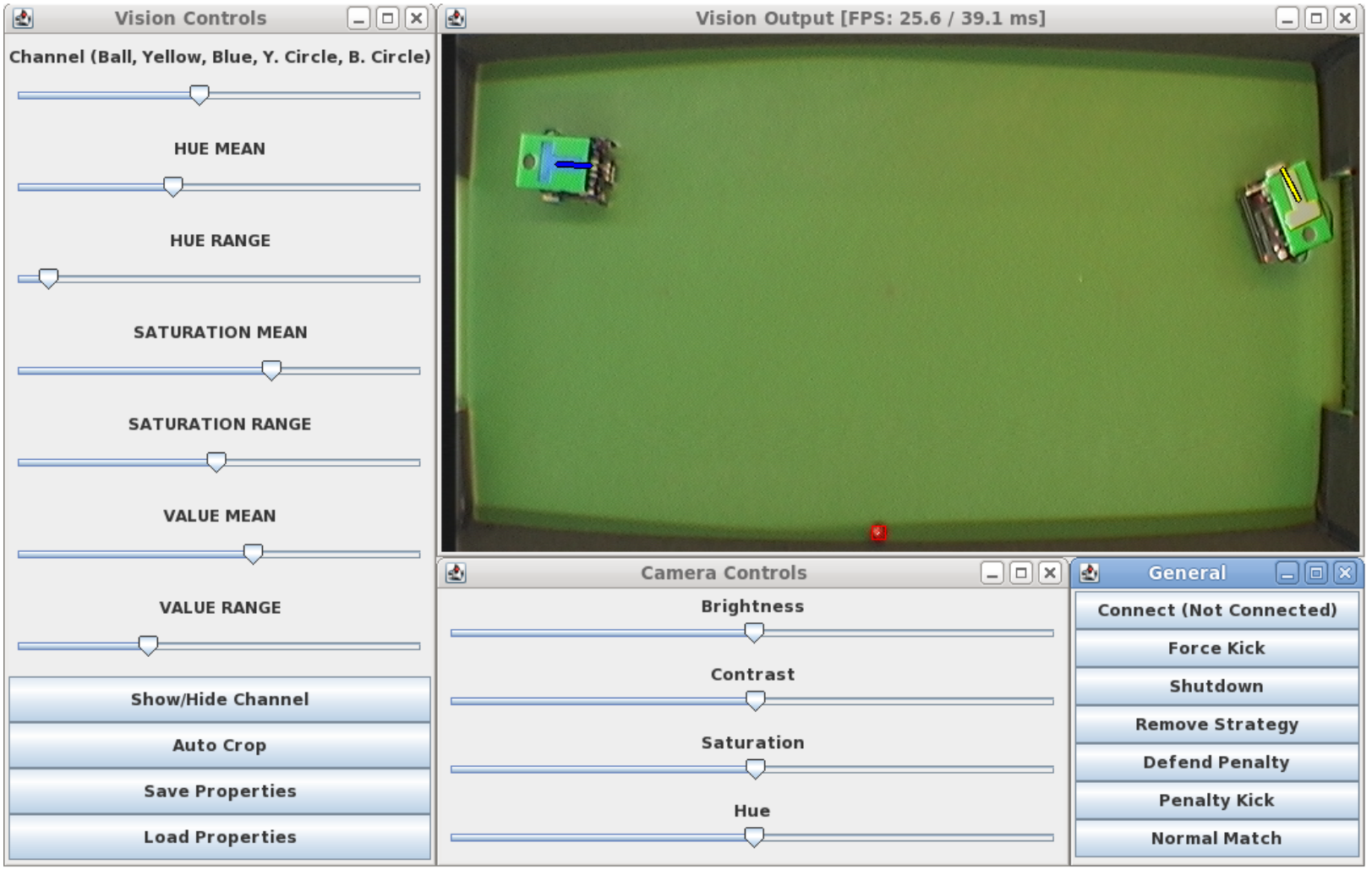
(200 hours * 5 persons) / 5 minutes =
200 person-hours / minute

Minimalism & Simplicity

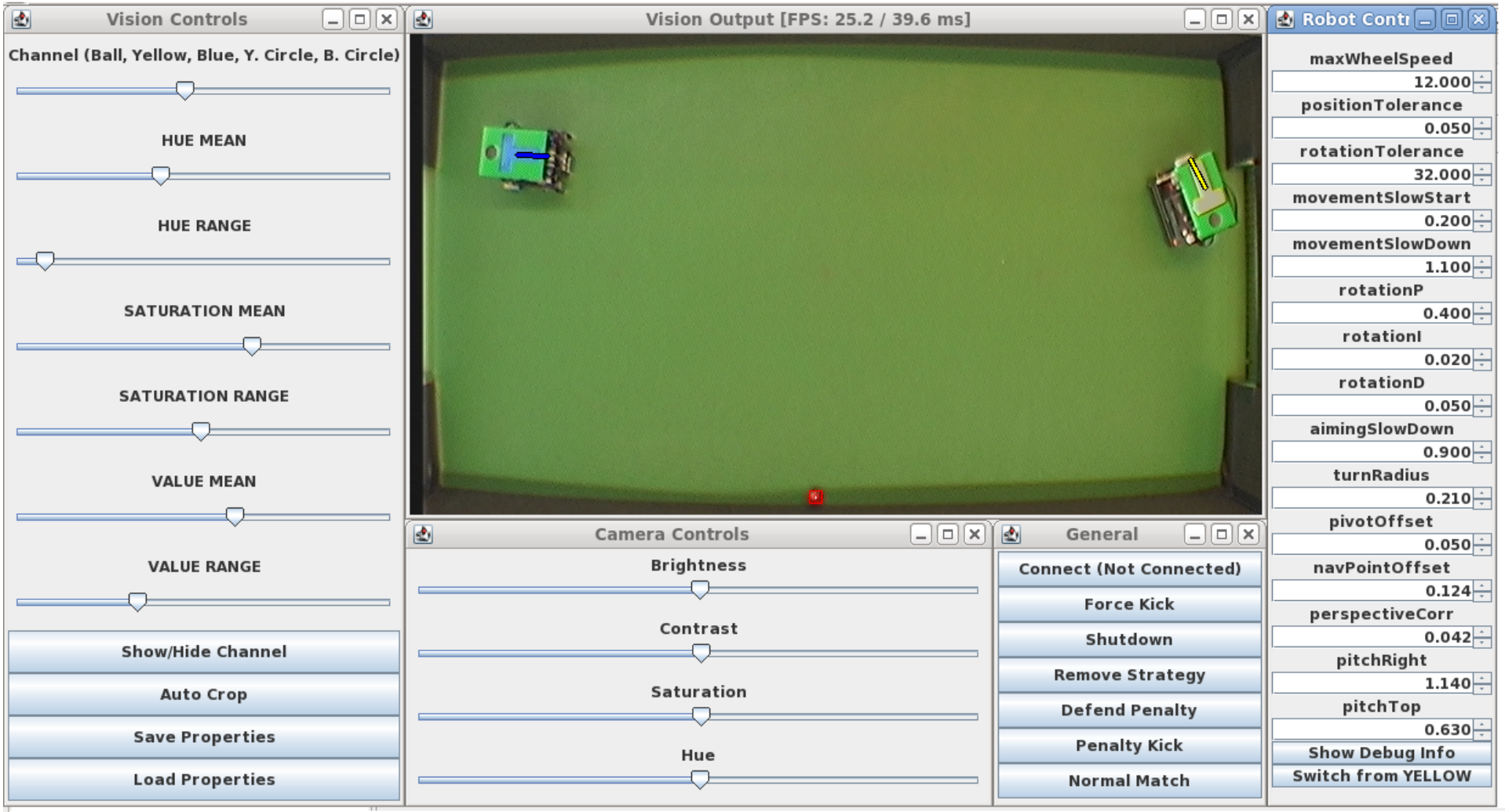
Minimalism & Simplicity



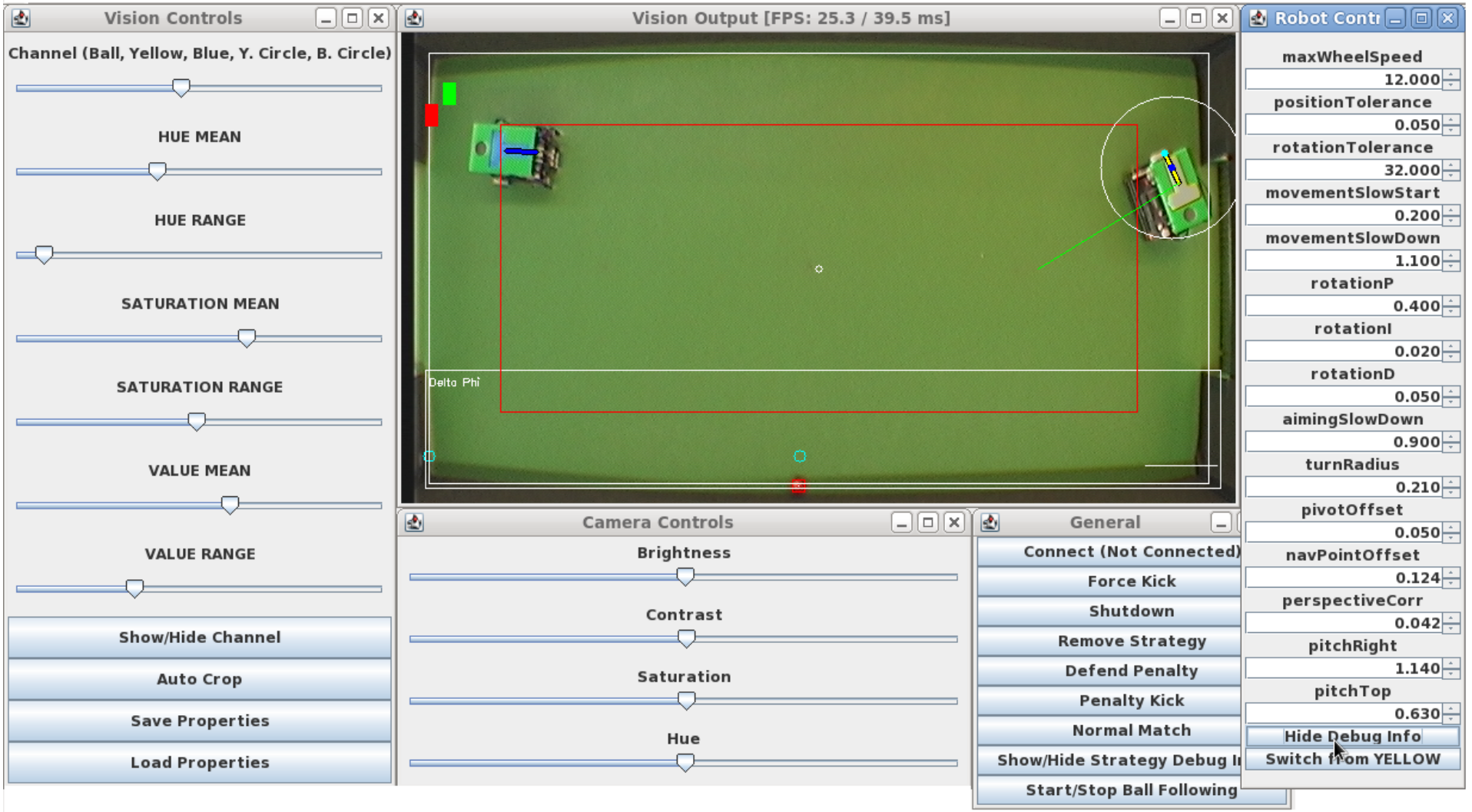
Minimalism & Simplicity



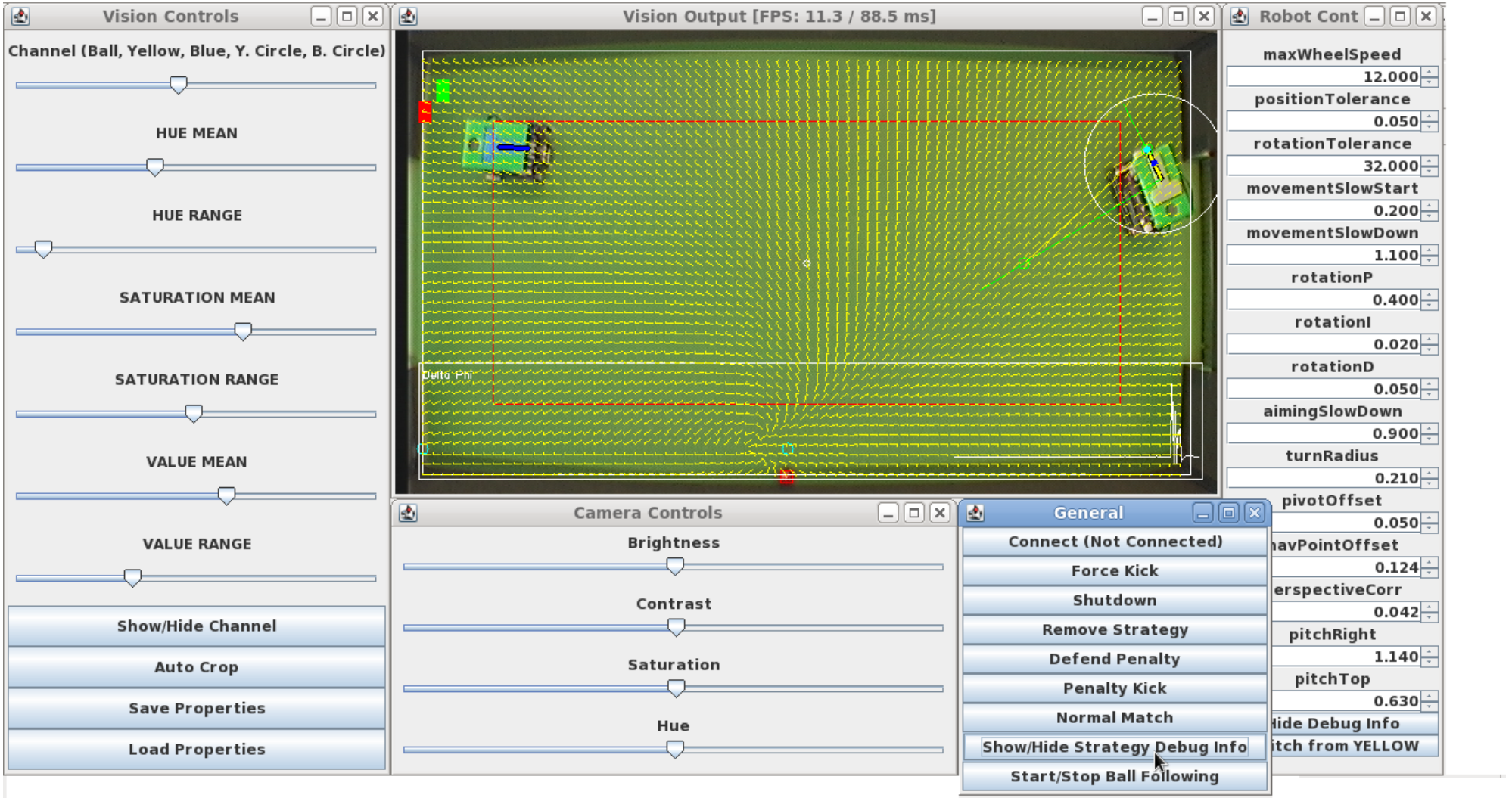
Minimalism & Simplicity



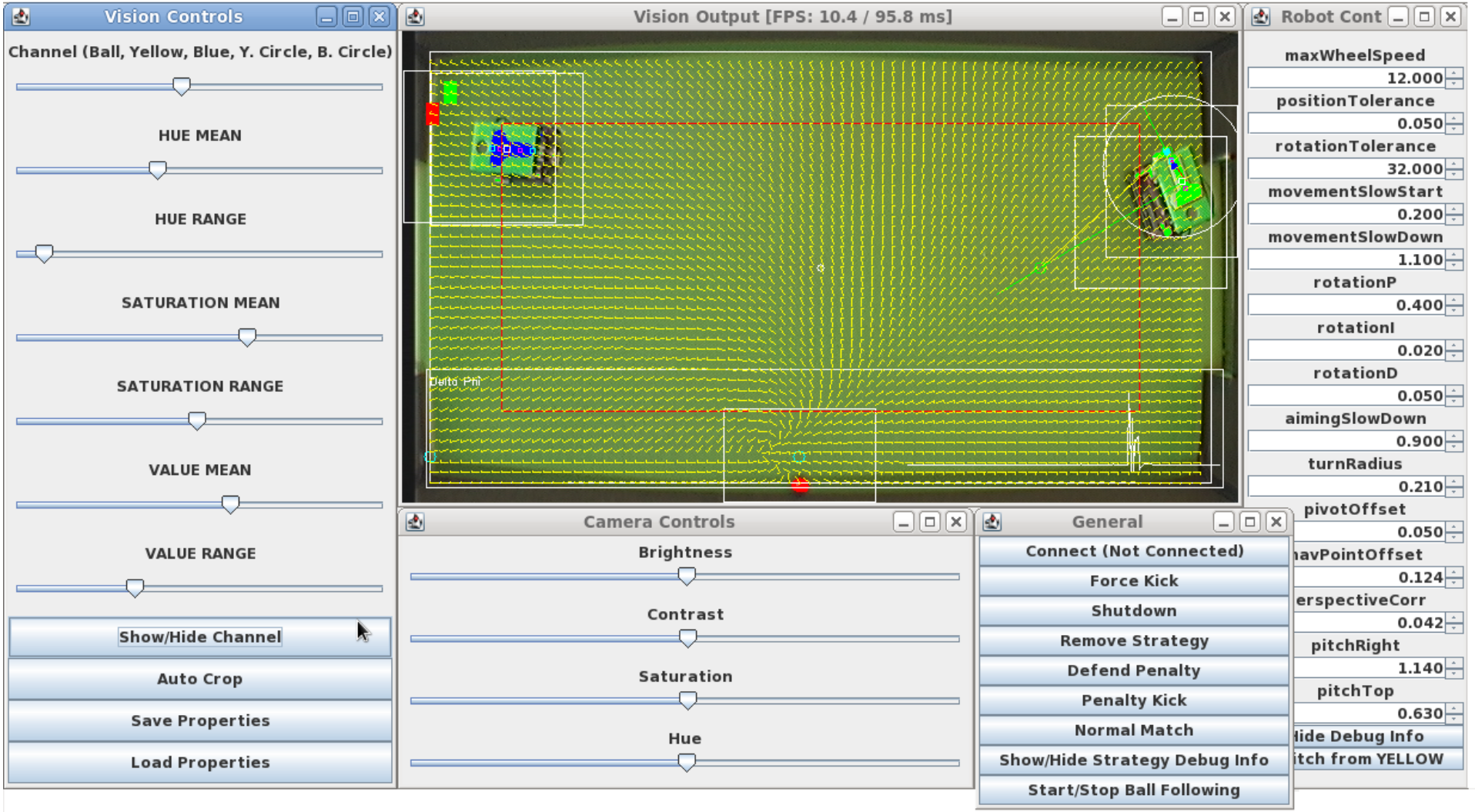
Minimalism & Simplicity



Minimalism & Simplicity



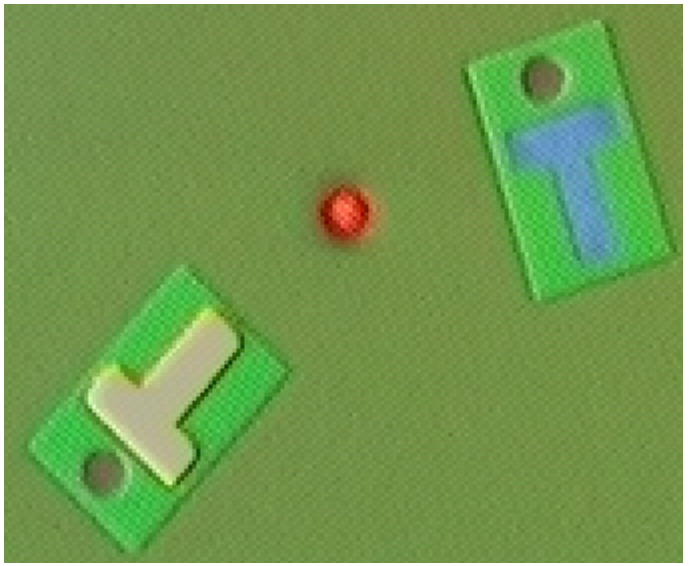
Minimalism & Simplicity



Vision: Goal

Vision: Goal

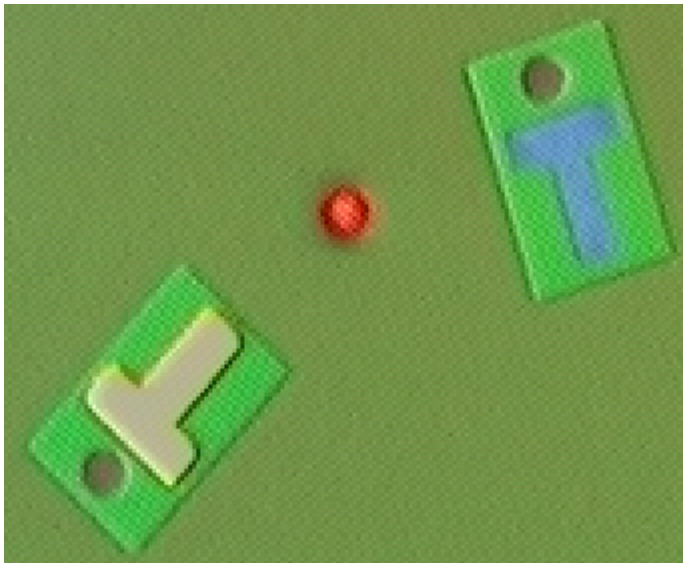
Original



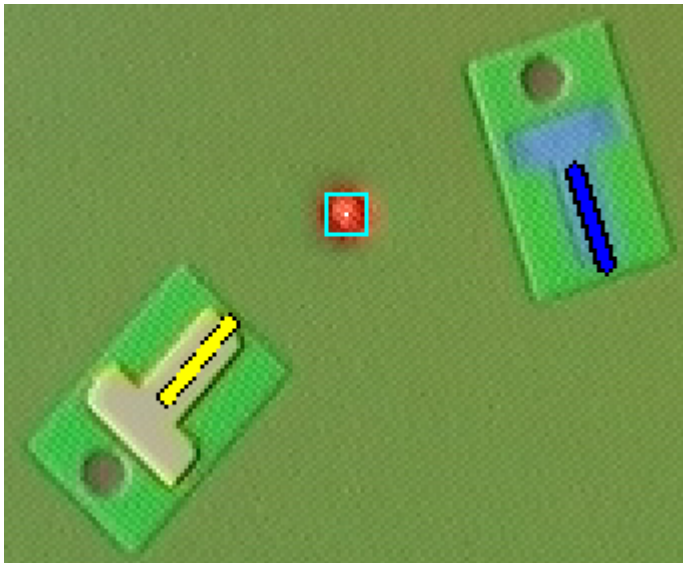
Estimated Pose

Vision: Goal

Original

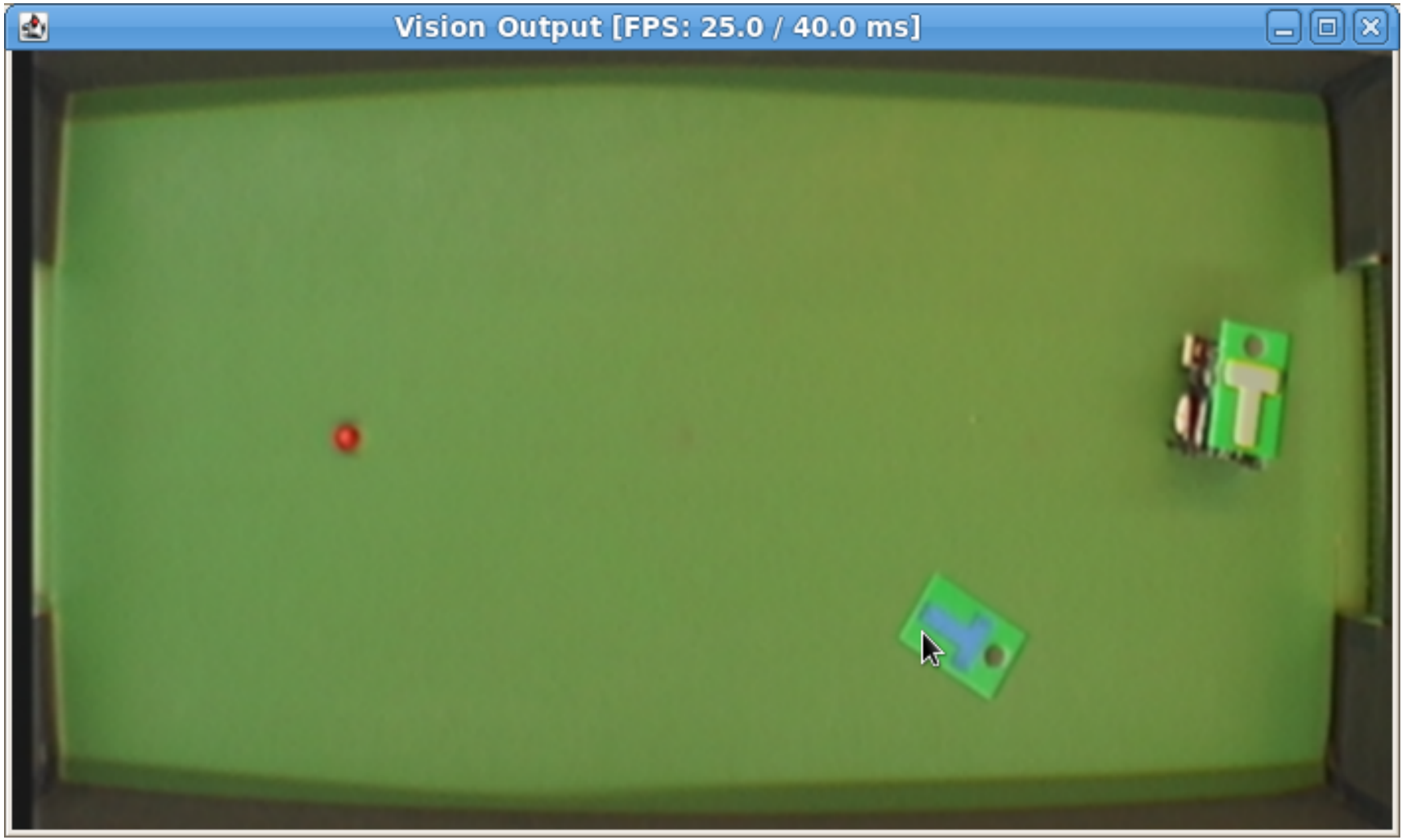


Estimated Pose

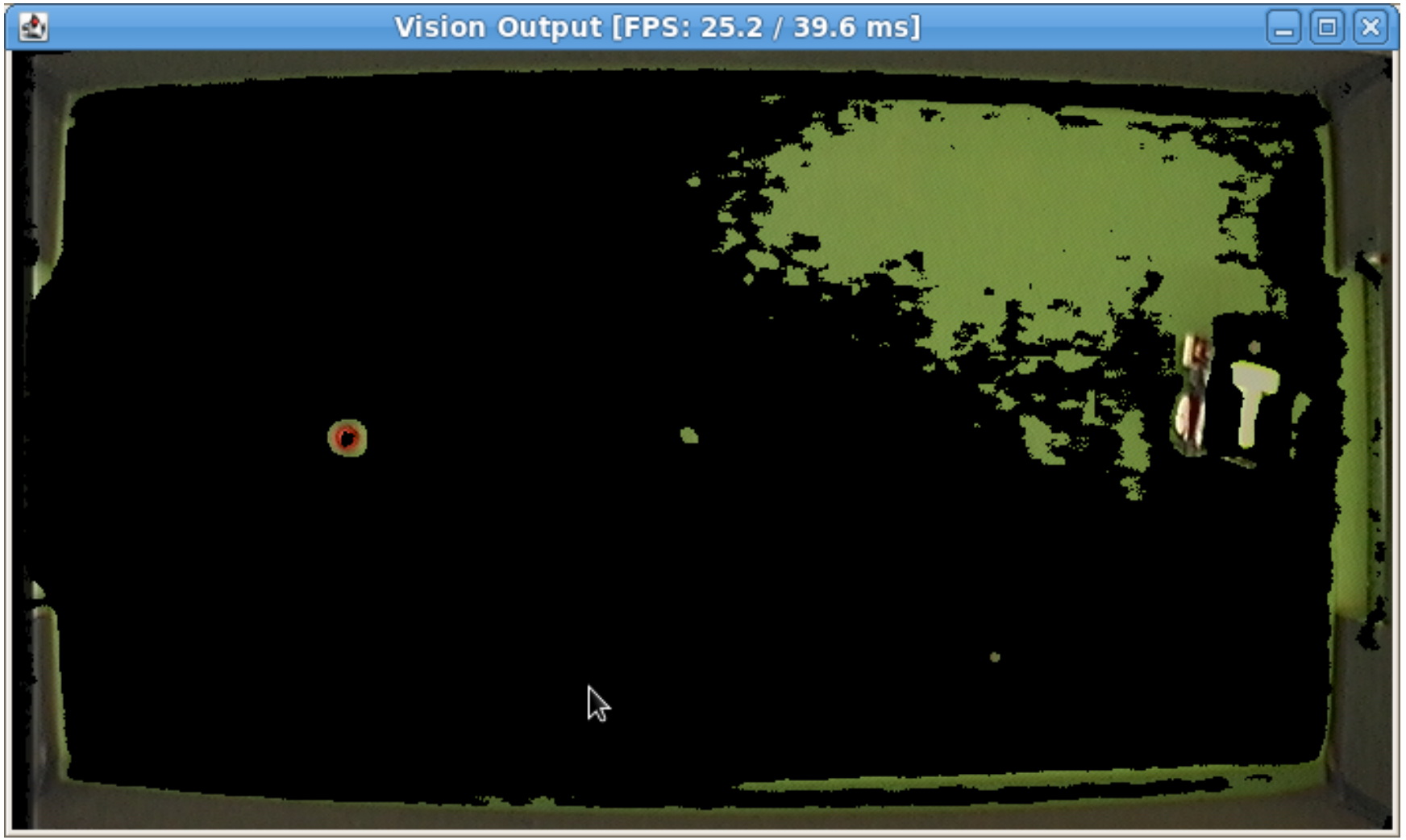


Vision: Detection - Raw

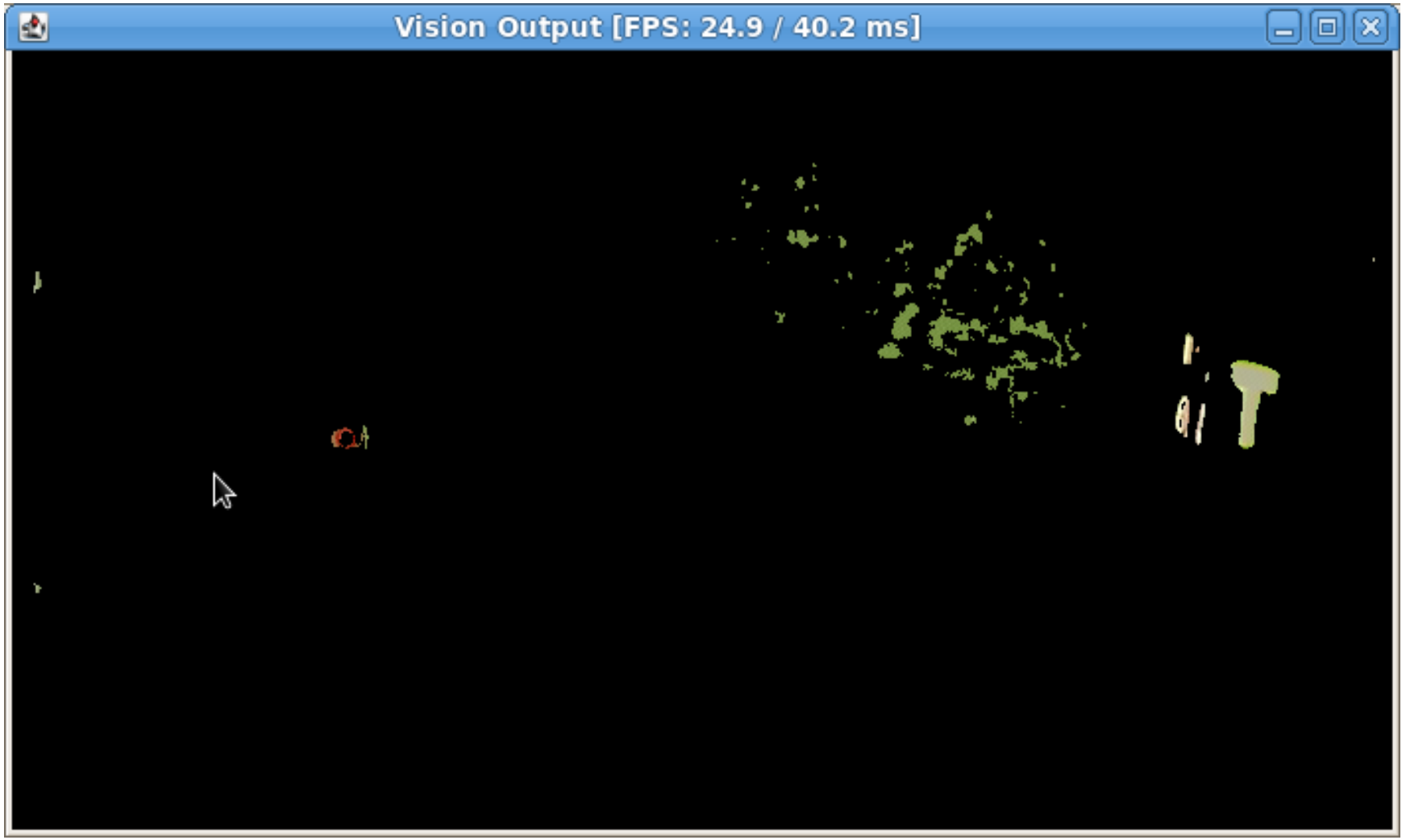
Vision: Detection - Preprocessed



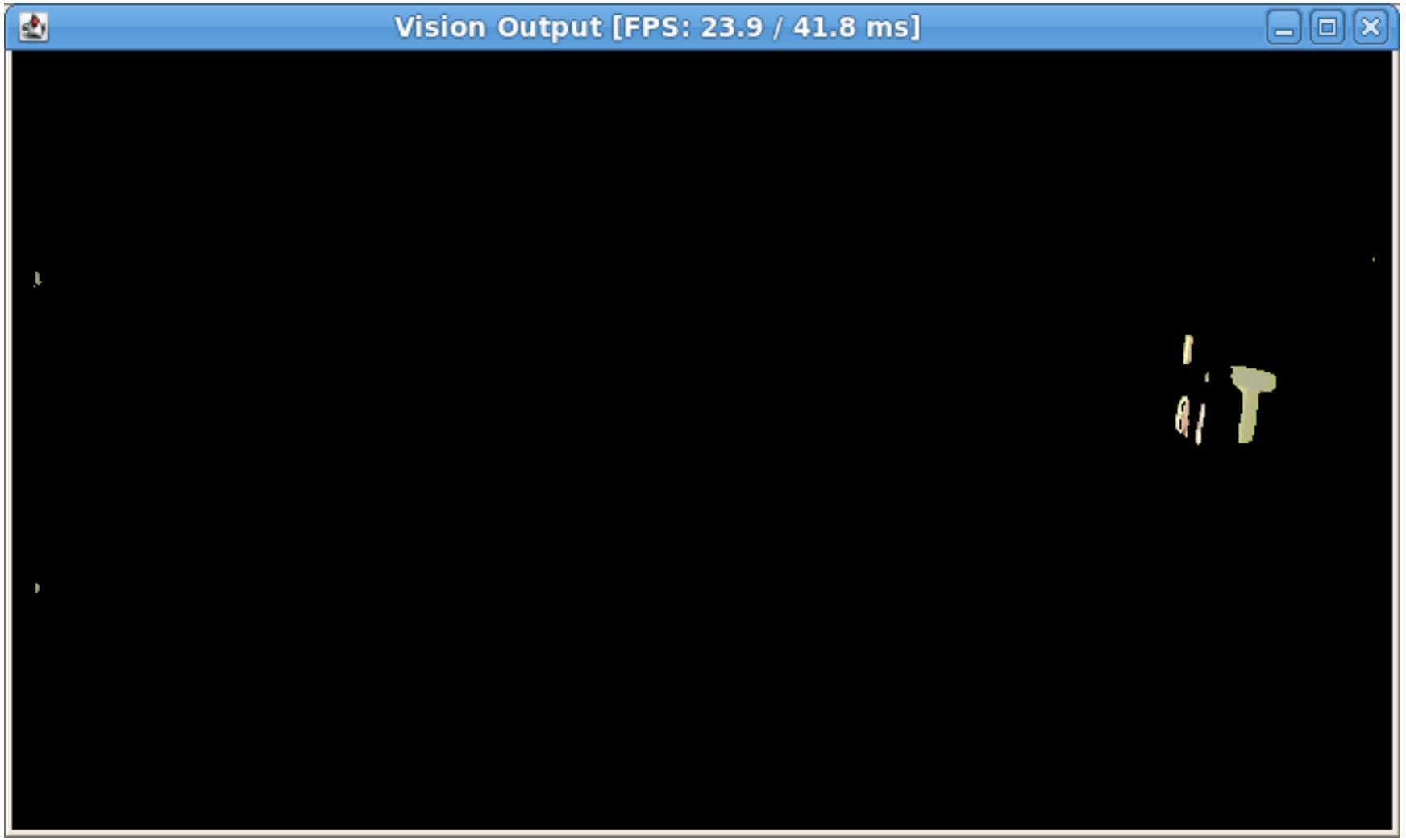
Vision: Detection – Hue



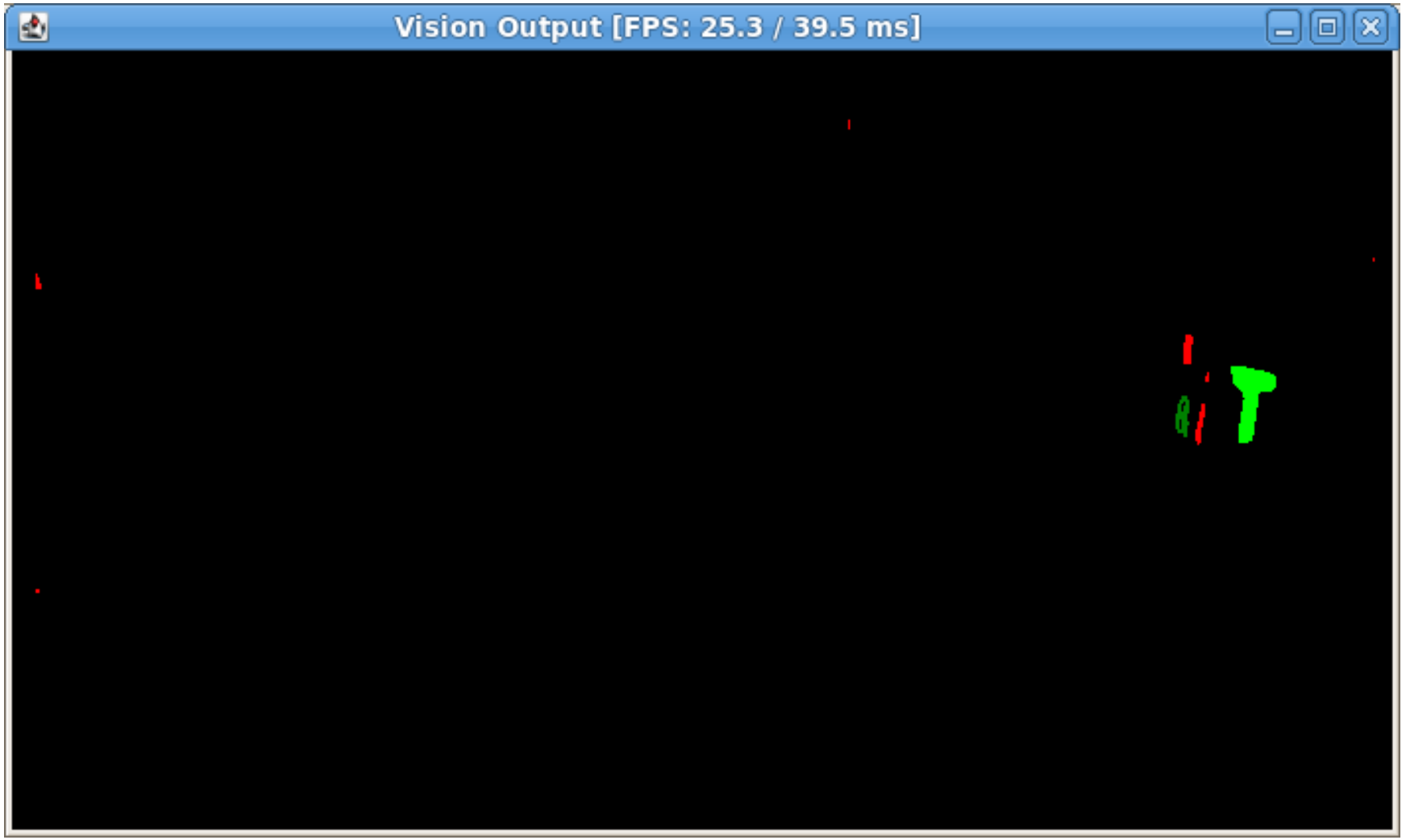
Vision: Detection – Hue + Value



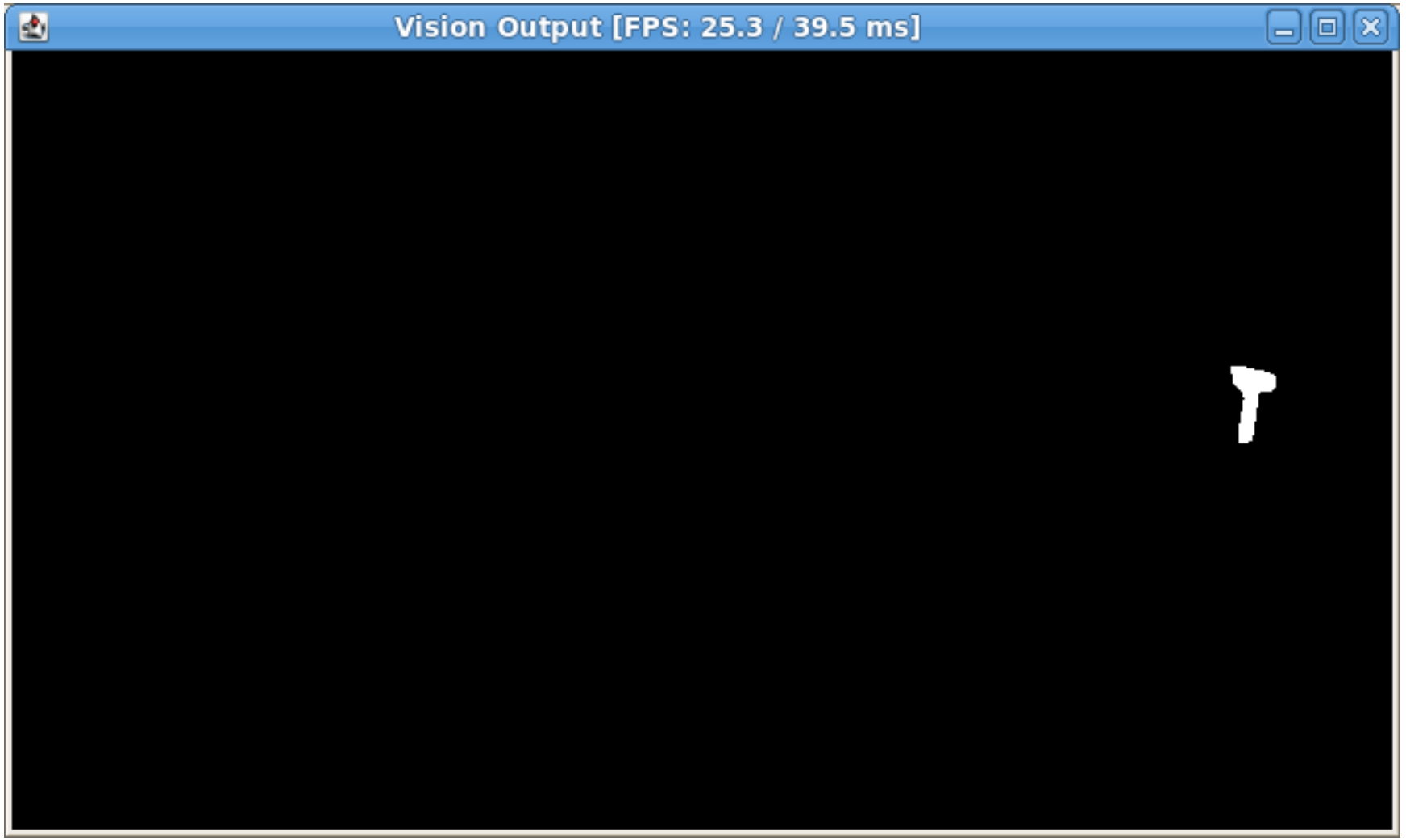
Vision: Detection – H, S and V



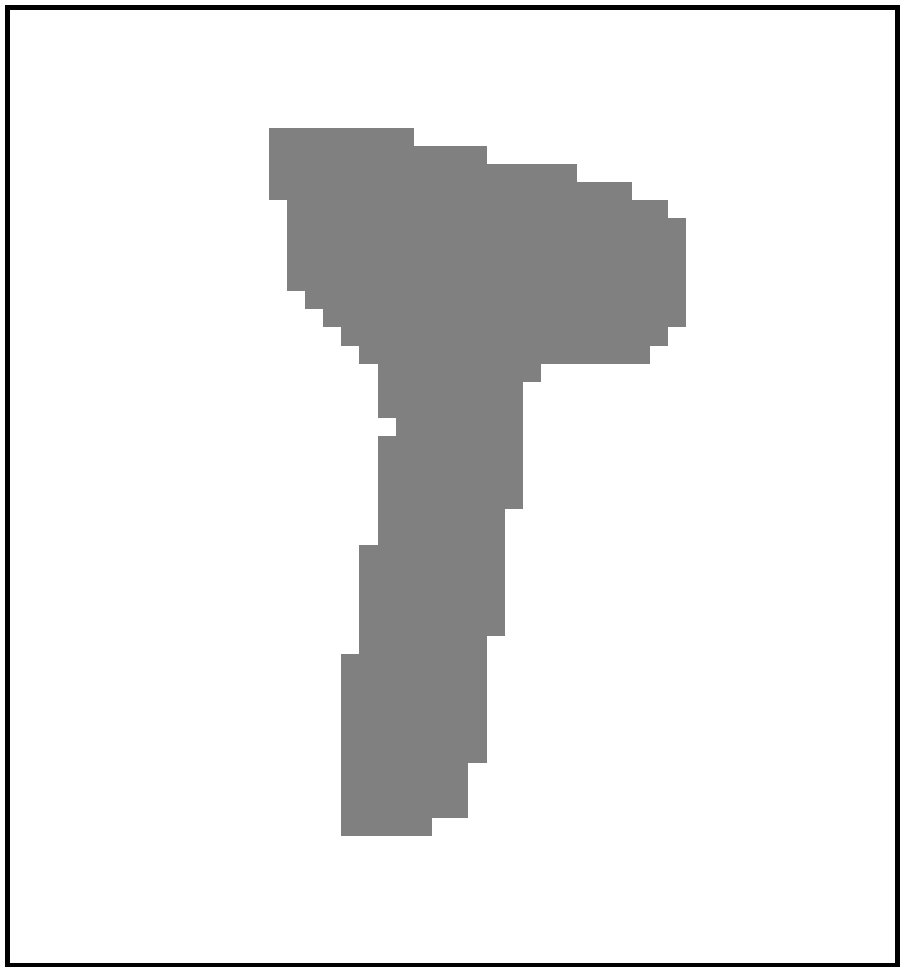
Vision: Detection – Shape Model



Vision: Detection – Final



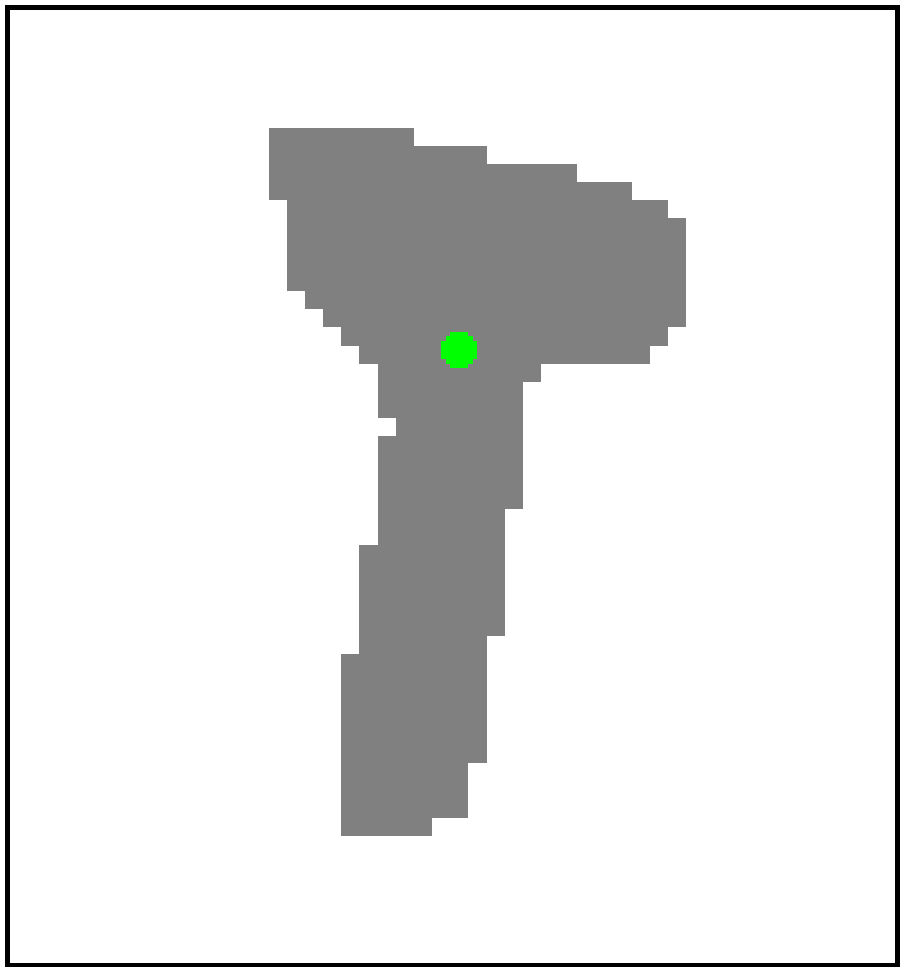
Vision: Pose Estimation



Estimate

- Position (2-D vector)
- Orientation (1 angle)

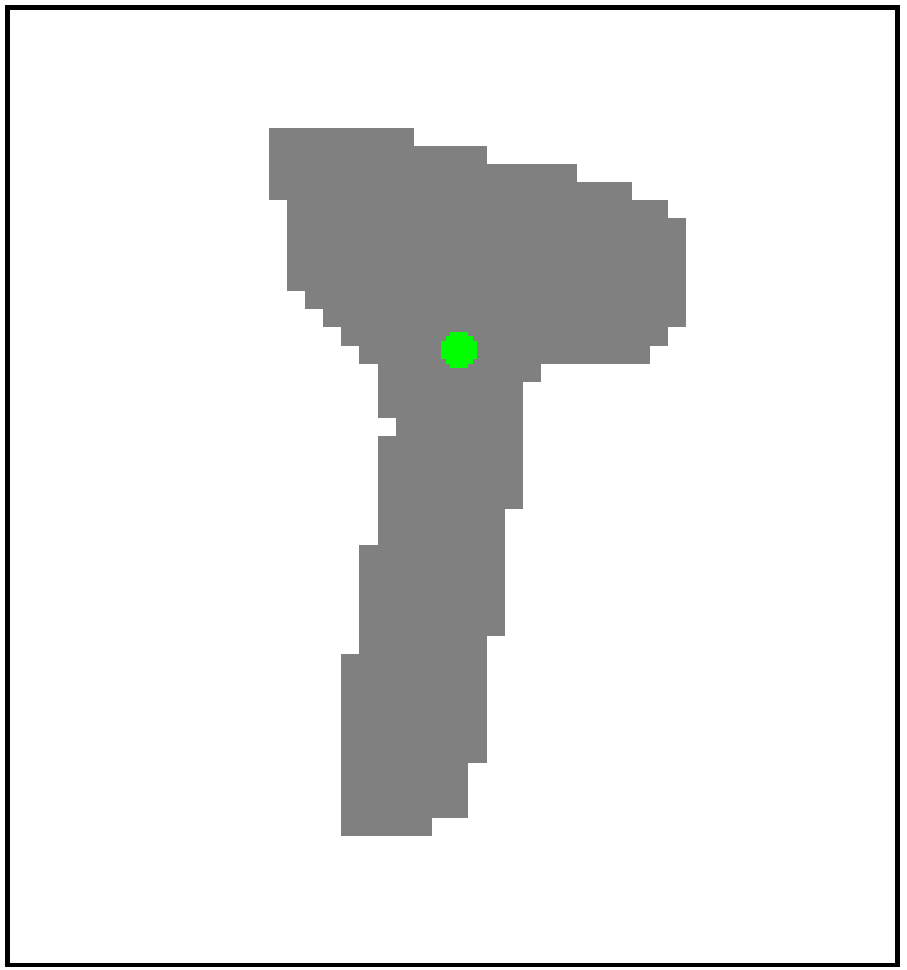
Vision: Pose Estimation



Position

- Centroid – average pixel coordinates

Vision: Pose Estimation

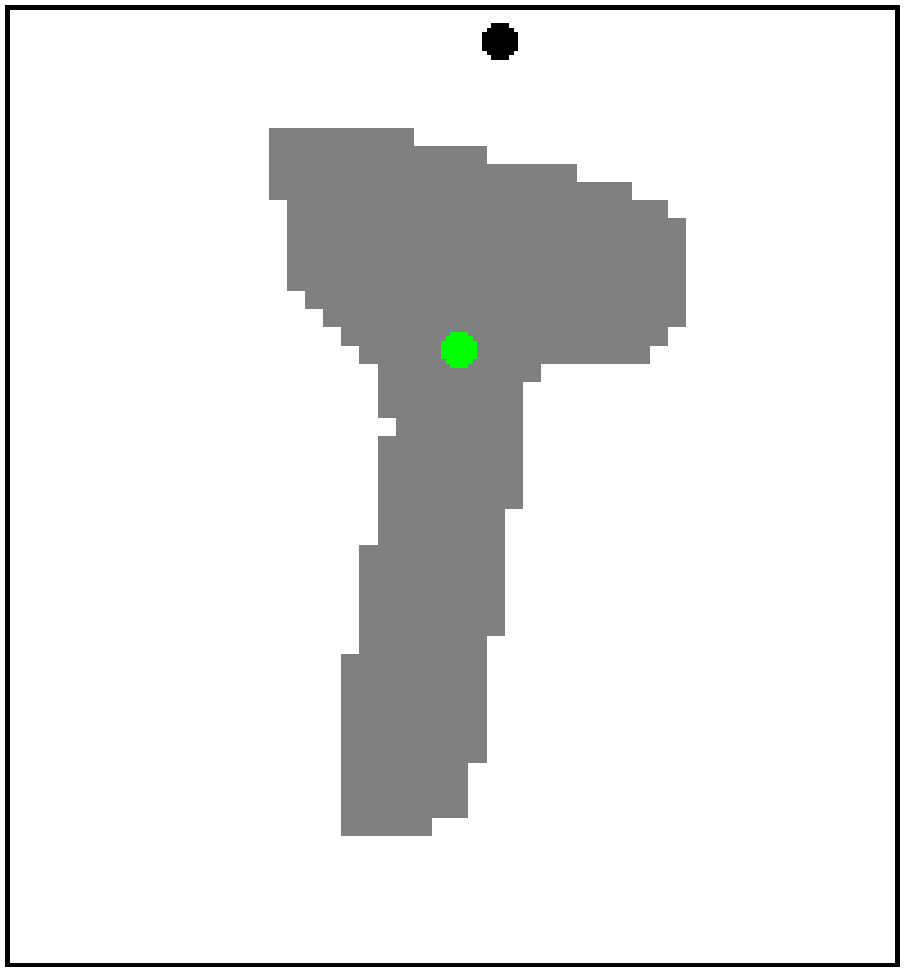


Orientation

Find points on line:

1. Use black circle.
2. Use convexity defects.

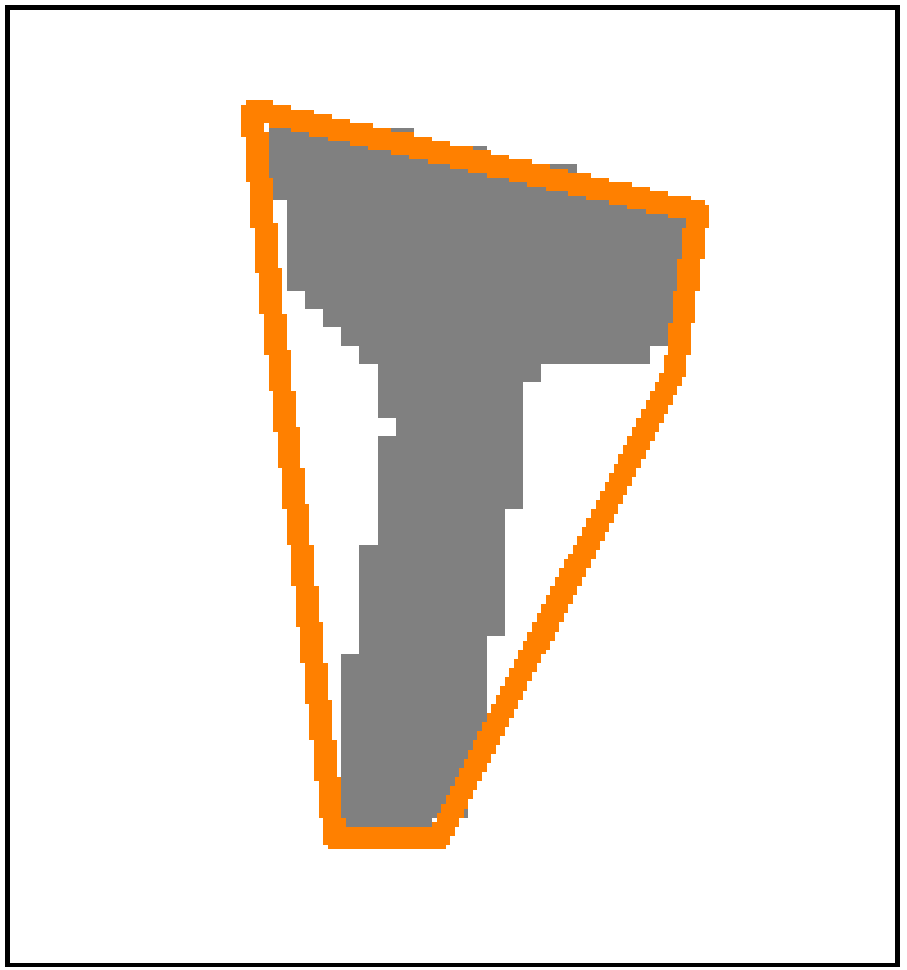
Vision: Pose Estimation



Orientation 1

- Detect black circle.
- Add black centroid to points.

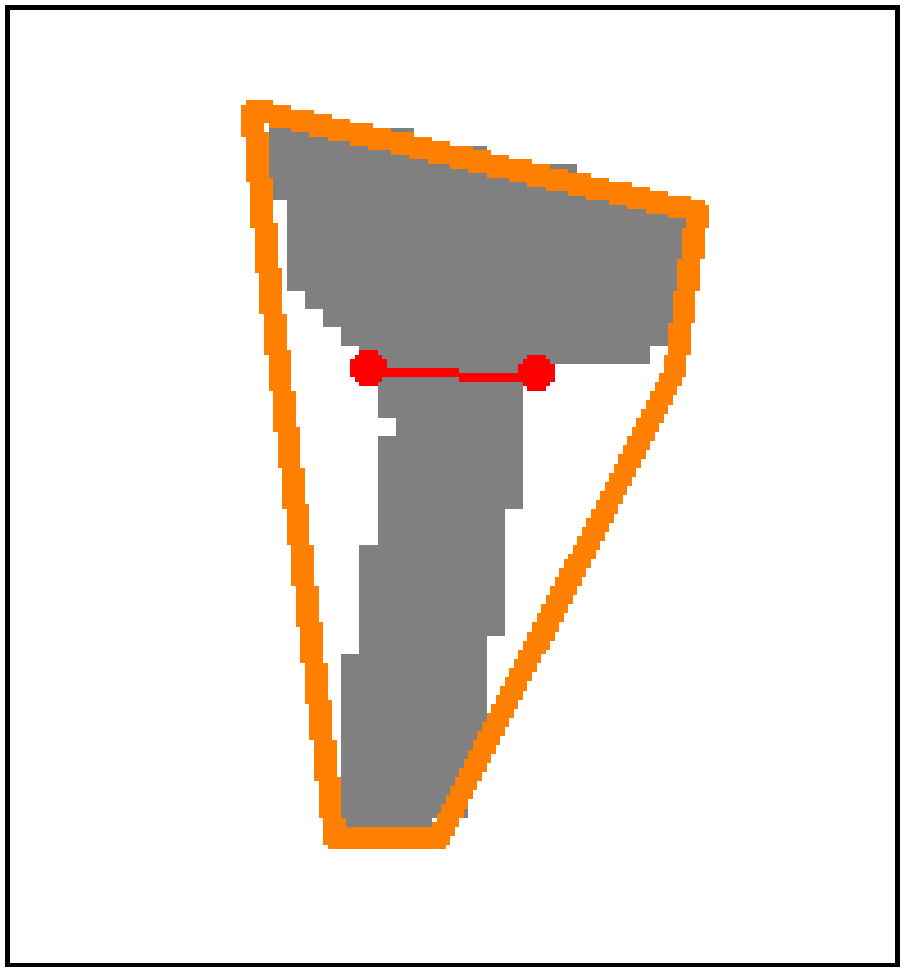
Vision: Pose Estimation



Orientation 2

- Compute convex hull.

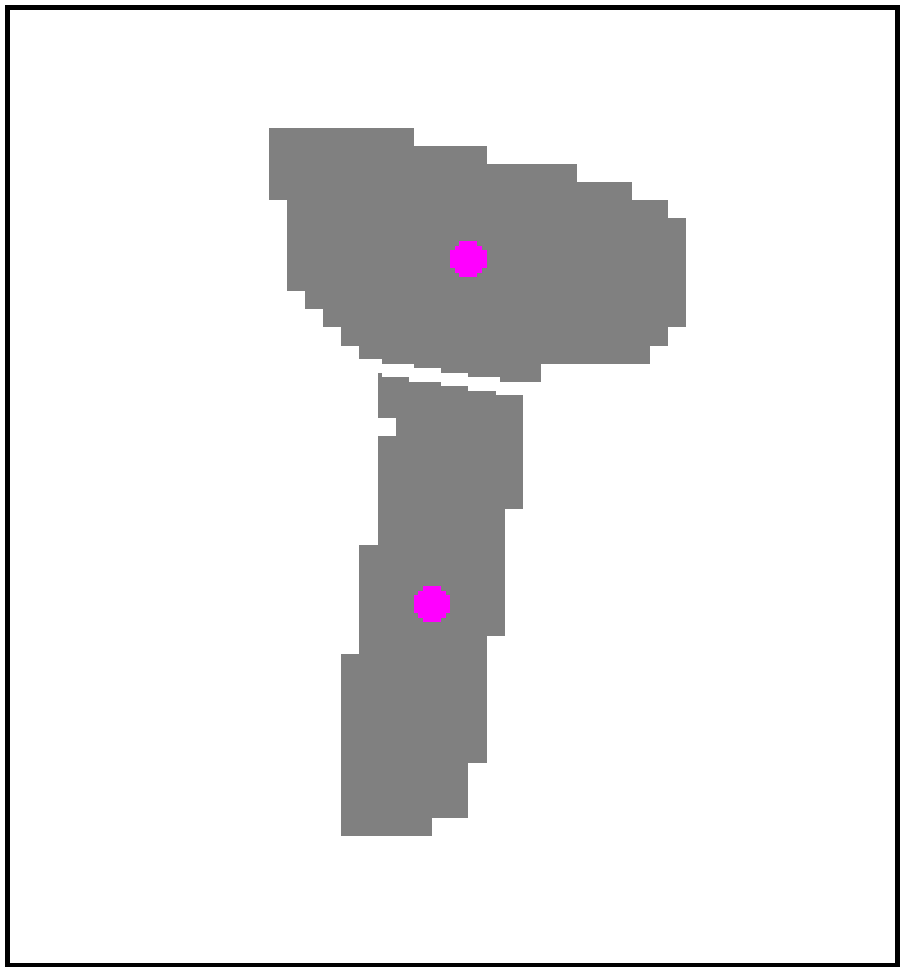
Vision: Pose Estimation



Orientation 2

- Compute convex hull.
- Find largest defects.

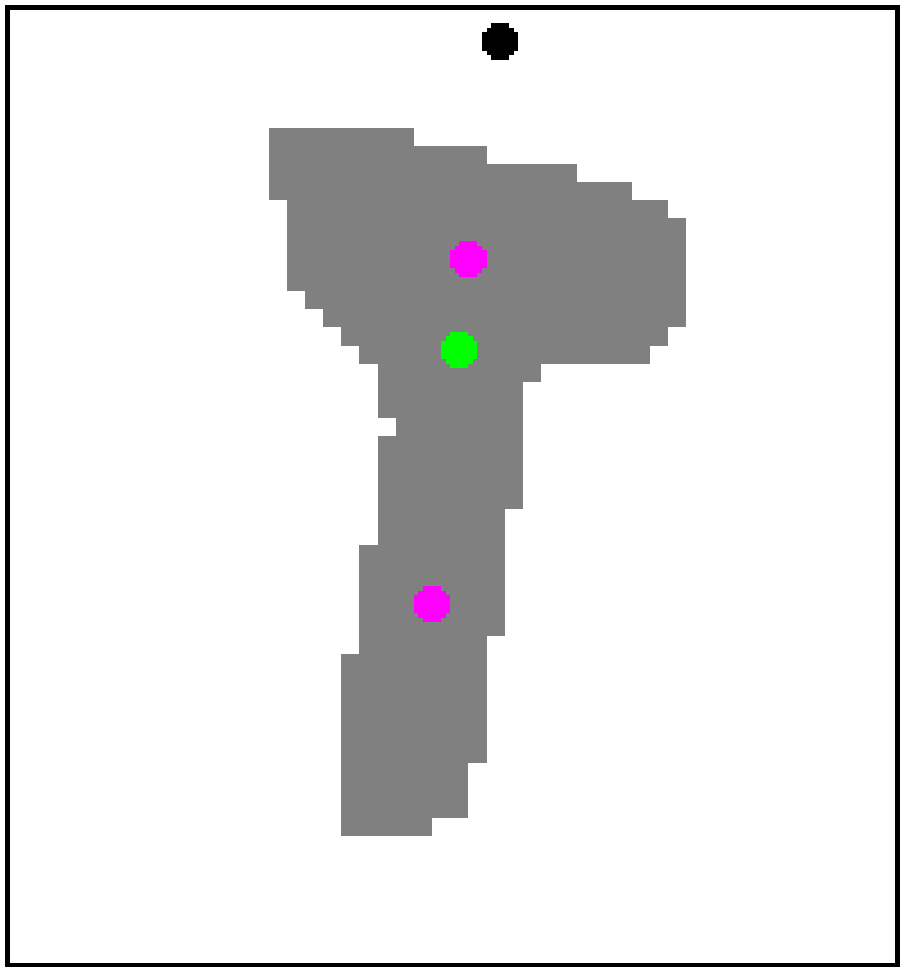
Vision: Pose Estimation



Orientation 2

- Compute convex hull.
- Find largest defects.
- Split along defect line, add sub-centroids to the points.

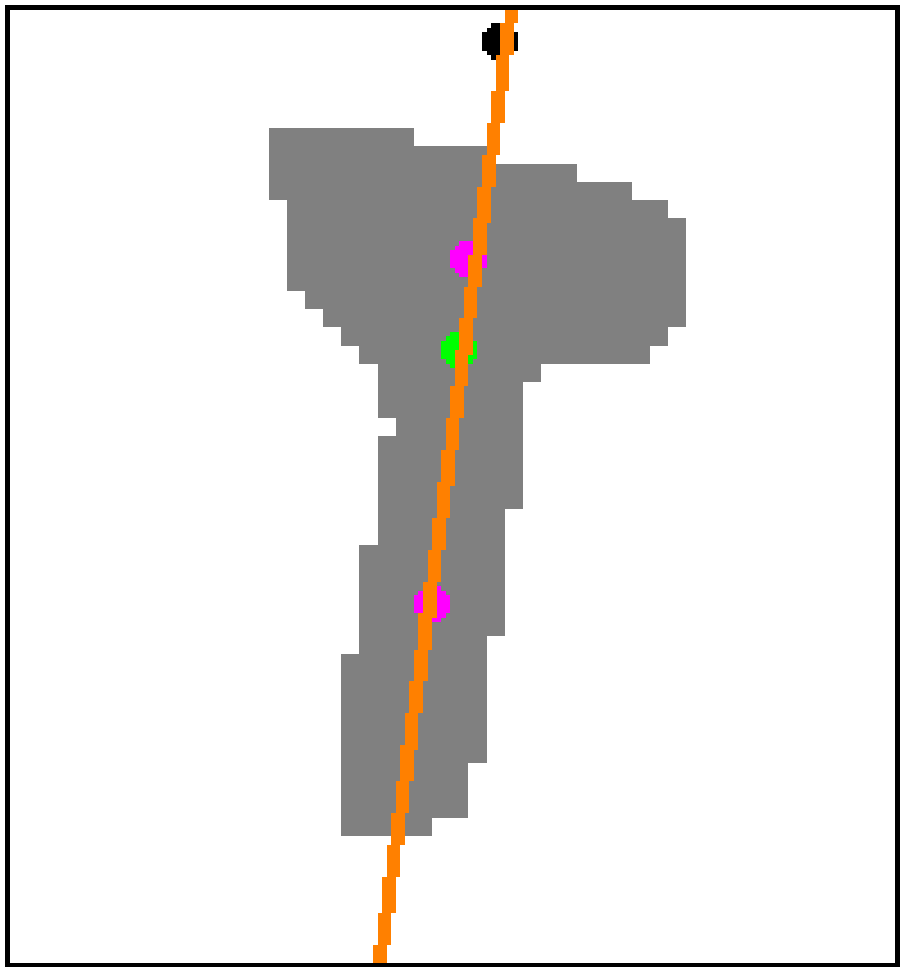
Vision: Pose Estimation



Orientation

- Four points that should lie on line.
- What method could we possibly use?

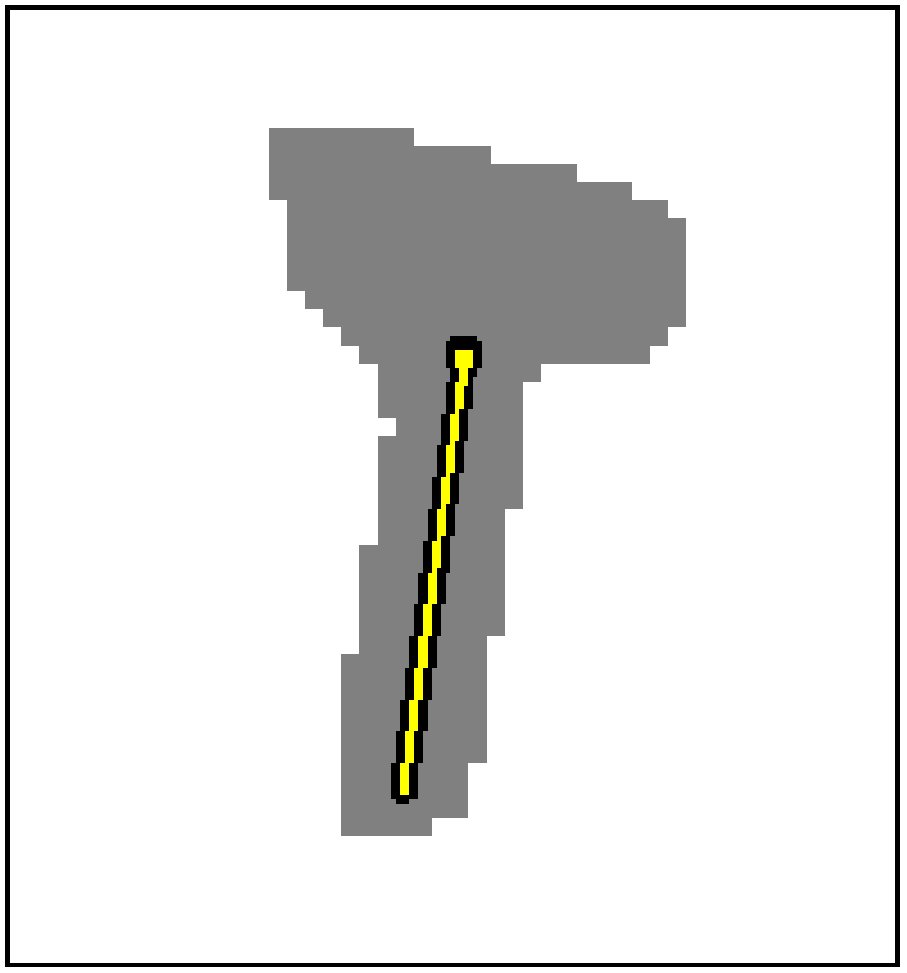
Vision: Pose Estimation



Orientation

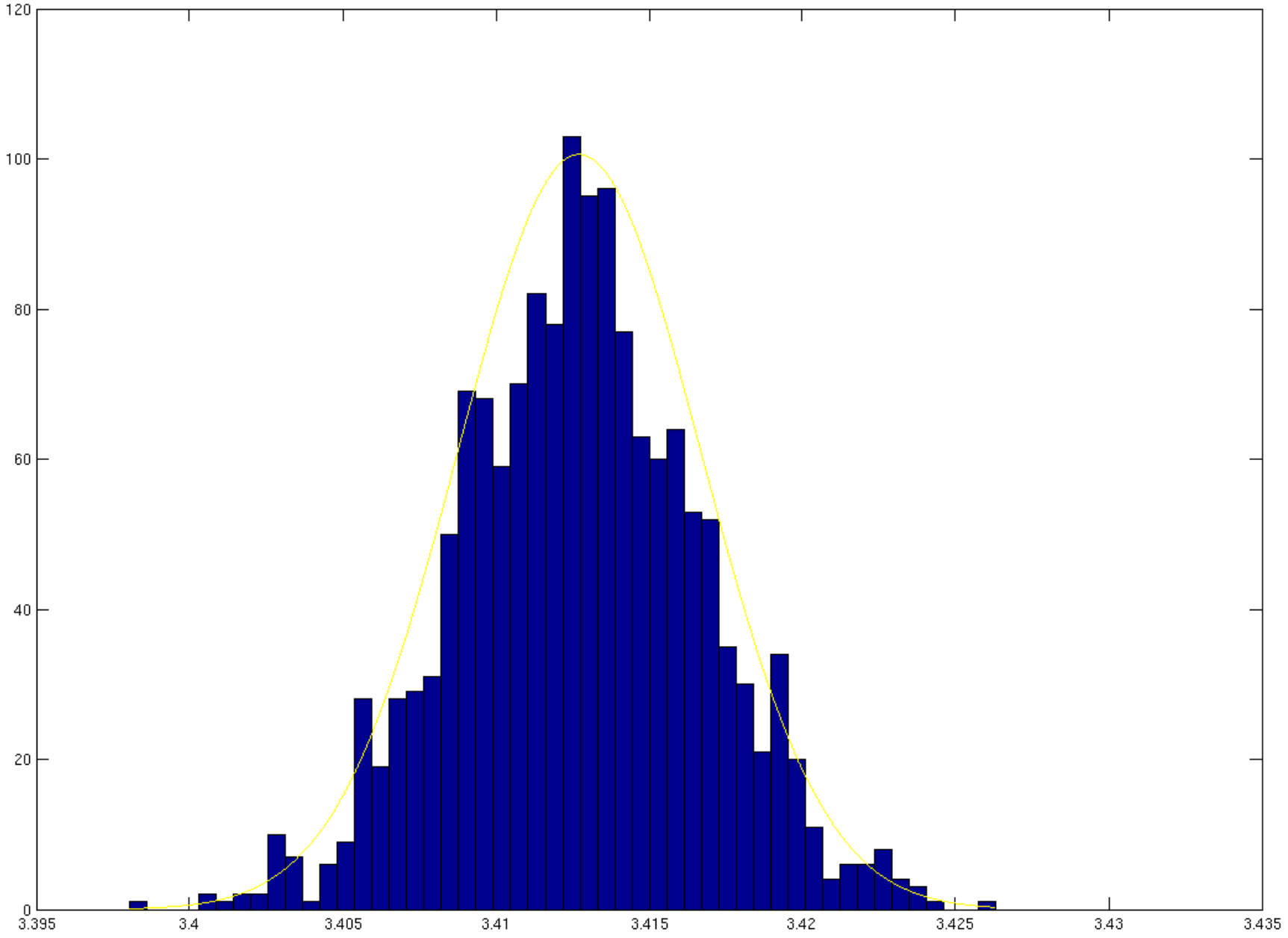
- Four points that should lie on line.
- Linear regression, of course.

Vision: Pose Estimation



Ta-da!

Vision: Estimation Error



Vision: Pose Estimation

Kalman Filters

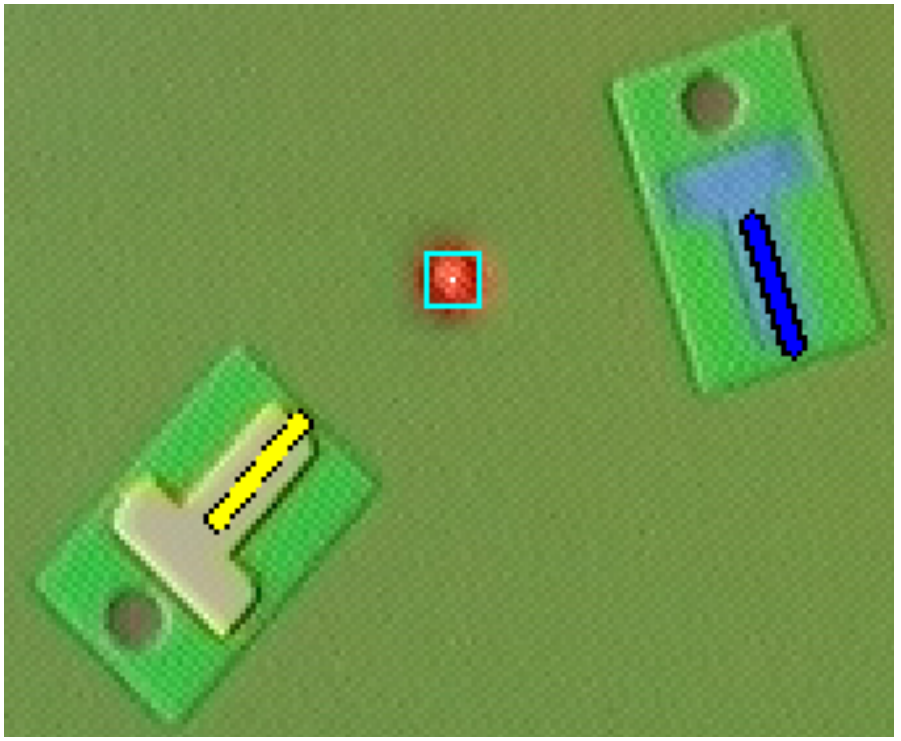
- Feed measurements into a Kalman filter.
- Use linear & angular velocity.

Vision: Pose Estimation

Kalman Filters

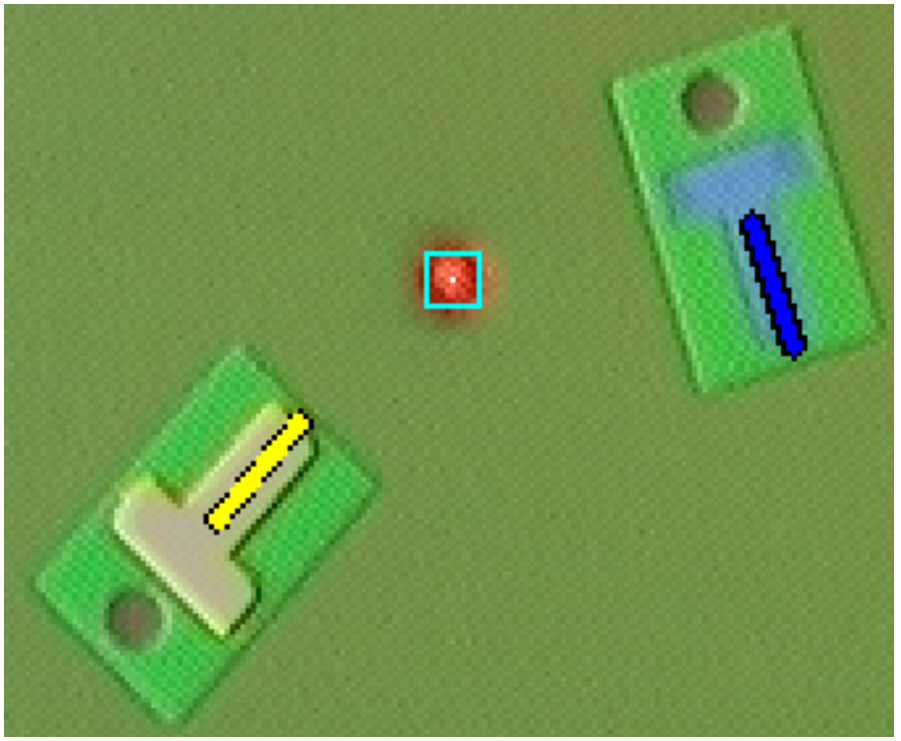
- Feed measurements into a Kalman filter.
- Use linear & angular velocity.
- Mr Kalman – most representative figure I could think of.

Vision: In the End



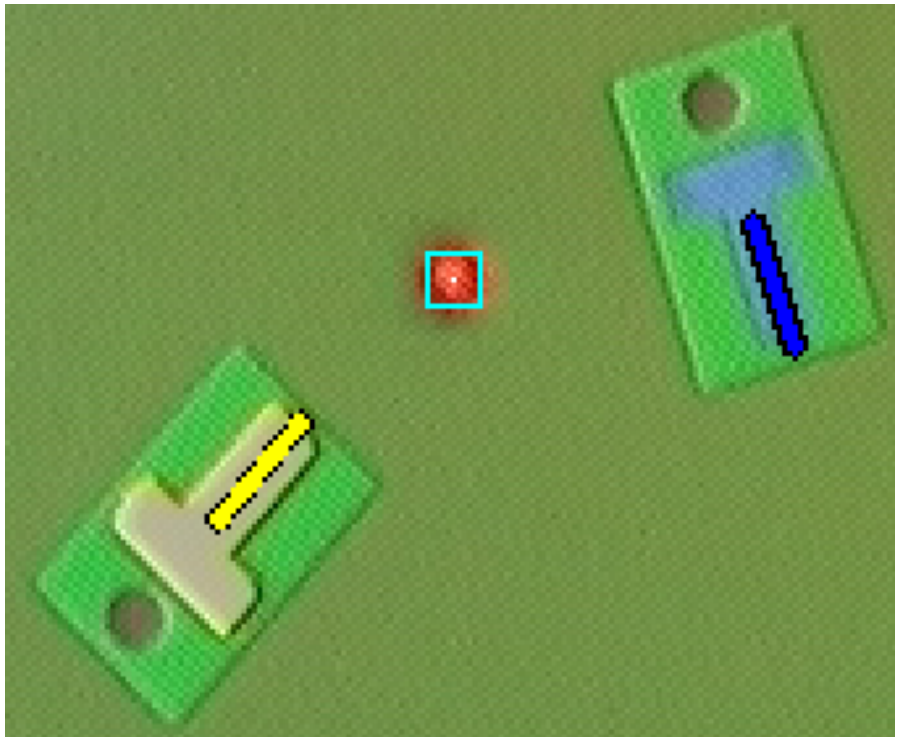
- Fast: ~25 FPS, limited by camera.

Vision: In the End



- Fast: ~25 FPS, limited by camera.
- Precise: ~2 degrees orientation error.

Vision: In the End



- Fast: ~25 FPS, limited by camera.
- Precise: ~2 degrees orientation error.
- Robust: multiple methods, one can fail – still works.