DRC Driving

Christopher Rasmussen
University of Delaware

February 15, 2013

Driving Contributors

- Drexel
  - Karthikeyan Yuvraj
  - Kiwon Sohn
  - Richard Vallett
  - Paul Oh
- Delaware
  - Qiaosong Wang
  - Brad Fletcher
  - Ioannis Poulakakis
  - Bert Tanner

Driving Task

From the latest version of the (virtual) competition rules:

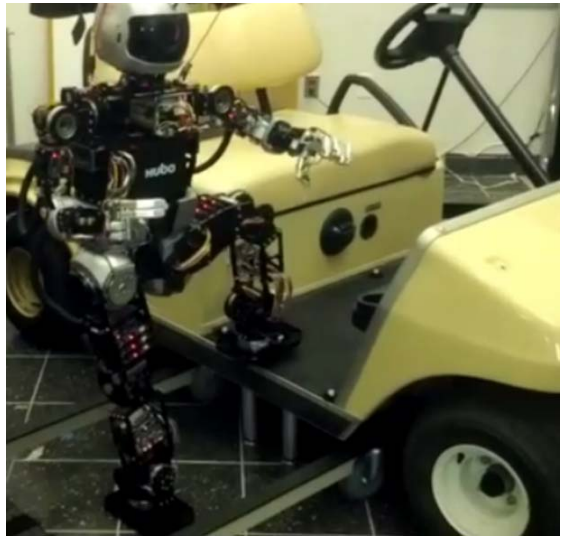
"Walk a short distance to and climb into a utility vehicle, drive along a roadway at no greater than 16 kph (10 mph), climb out of the utility vehicle, and walk to the finish area.

Like an arcade racing game, the scoring methodology provides rewards upon completion of each checkpoint*...Teams start a run with fixed resources -- time, uplink bits, and downlink bits*

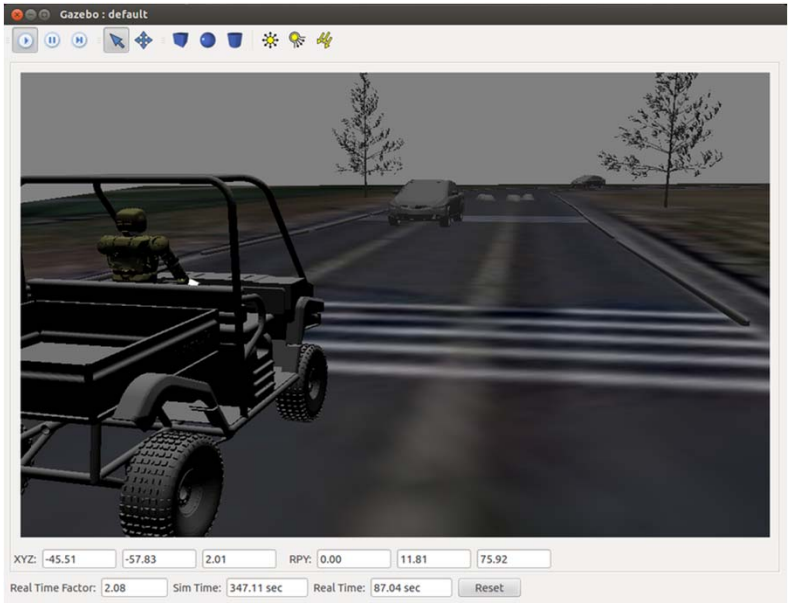
Checkpoints for the driving task will be placed ...such that 20% of the checkpoints are from the initial walking / ingress sub-task, 60% of the checkpoints concern the driving sub-task, and 20% of the checkpoints surround the egress and final walking sub-task..."

Driving Phases

- Ingress/egress
  - Approach
  - Step up
  - Sit down
  - Interface with controls



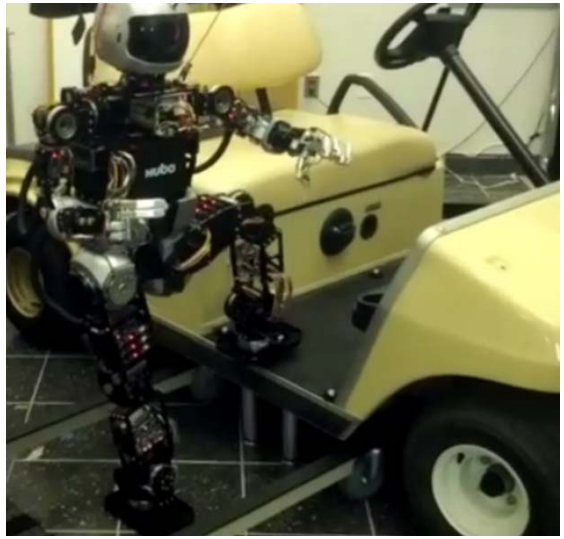
- Driving
  - Calibration of controls
  - Obstacle detection
  - Road tracking
  - Motion planning
  - Long range navigation



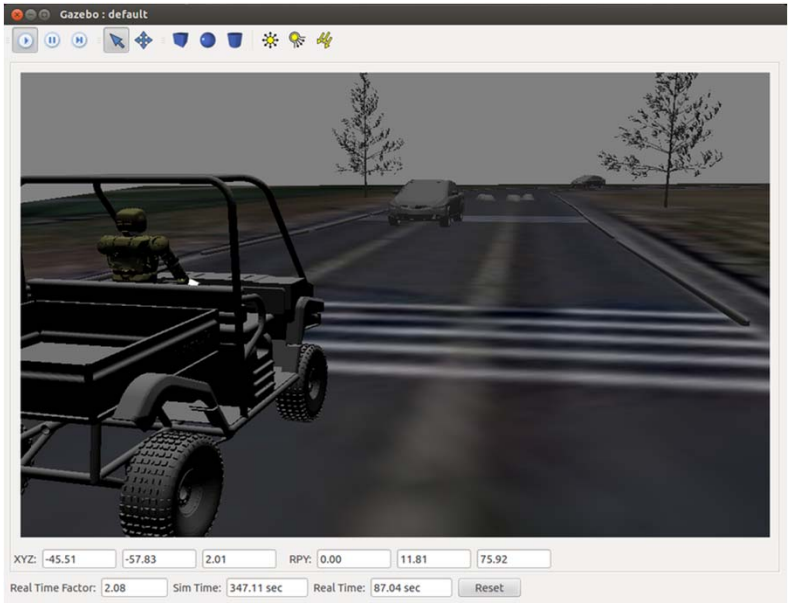
Progress in Brief

Little to no work done yet
Some work and testing done
Complete or almost done

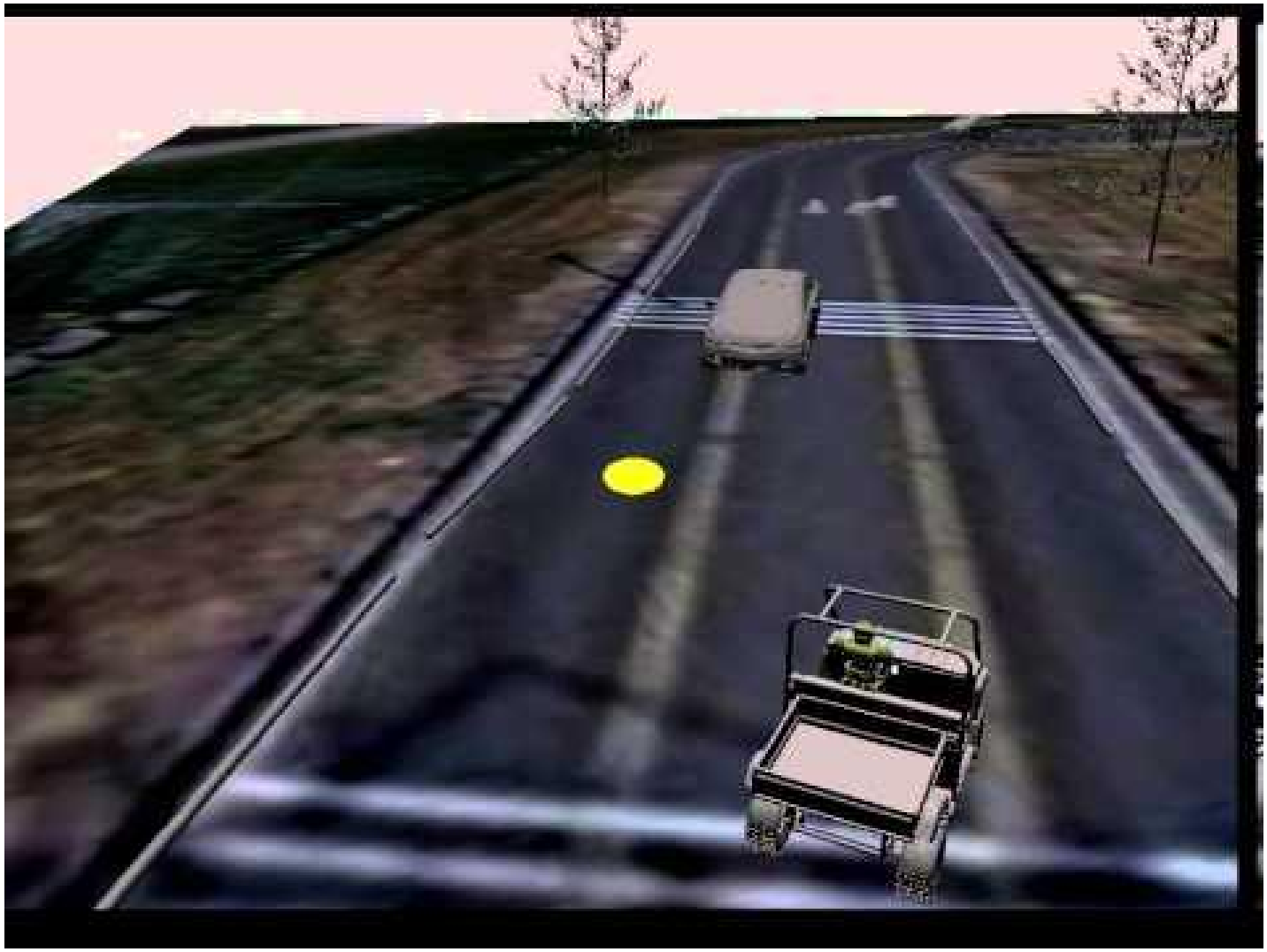
- Ingress/egress
  - Approach
  - Step up
  - Sit down
  - Interface with controls



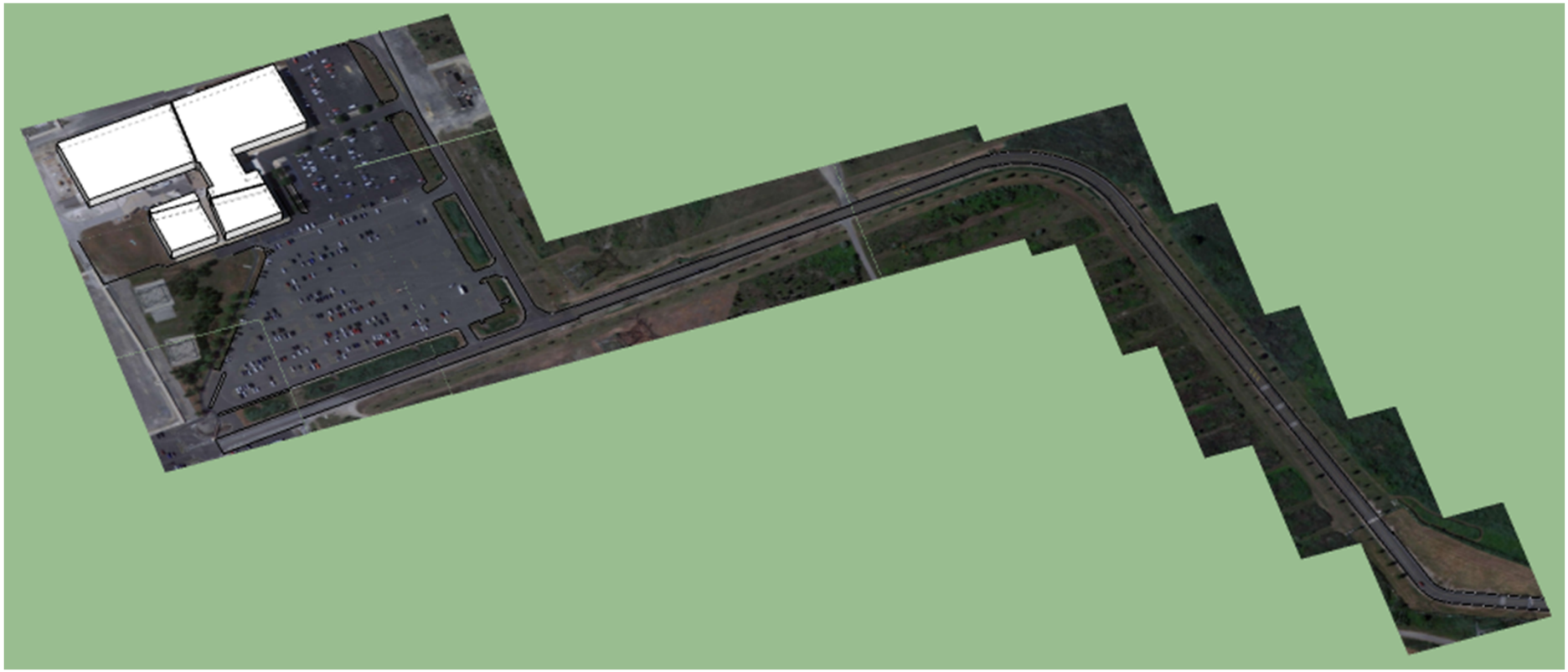
- Driving
  - Calibration of controls
  - Obstacle detection
  - Road tracking
  - Motion planning
  - Long-range navigation



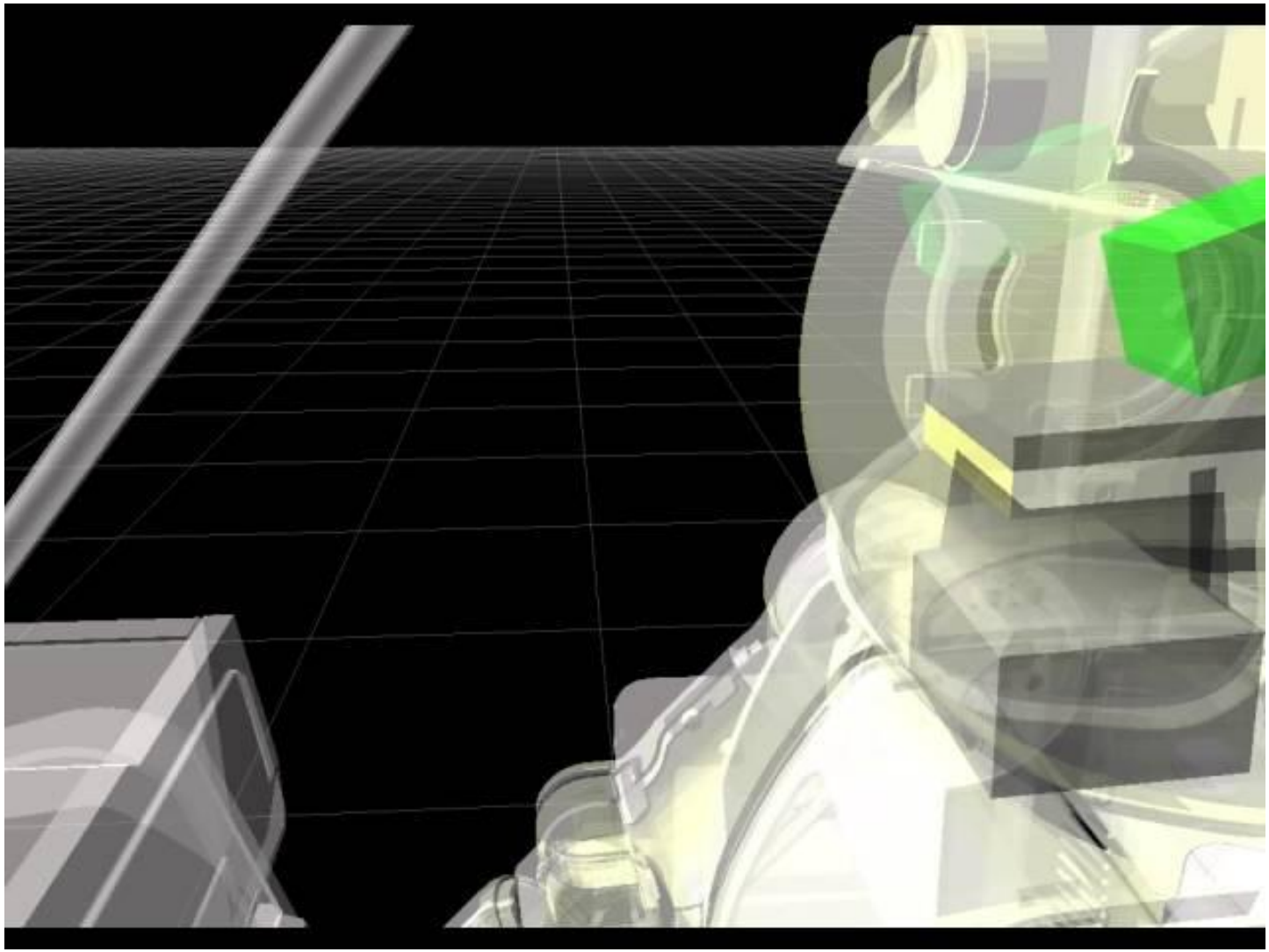
Simulated Driving in Gazebo



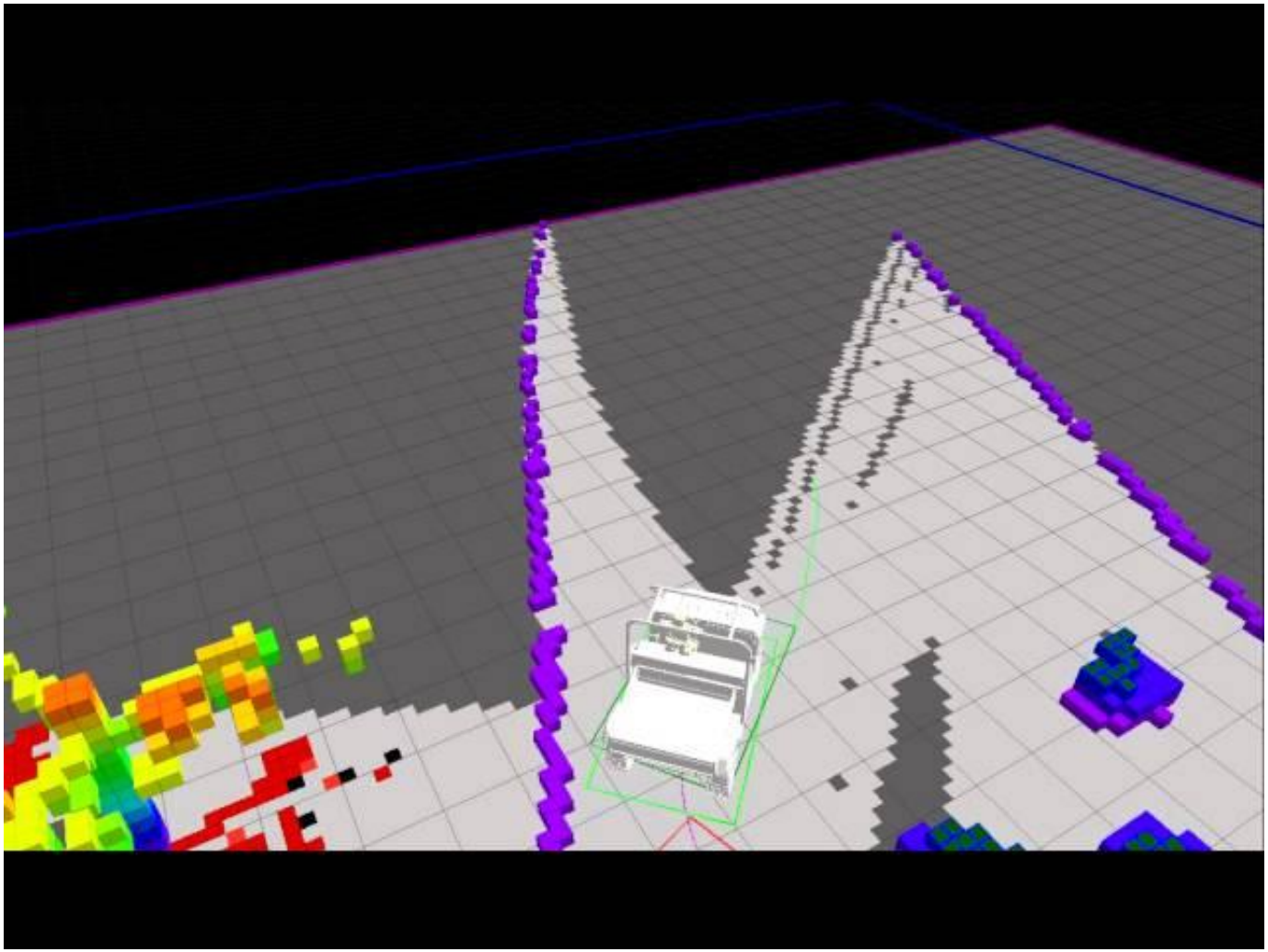
Simulated Driving Route #1: Salem



Tilting Ladar in Gazebo



Mapping, Motion Planning in Gazebo



Waypoints used here for long-range navigation

Prototype Driving Sensor Head v1.1

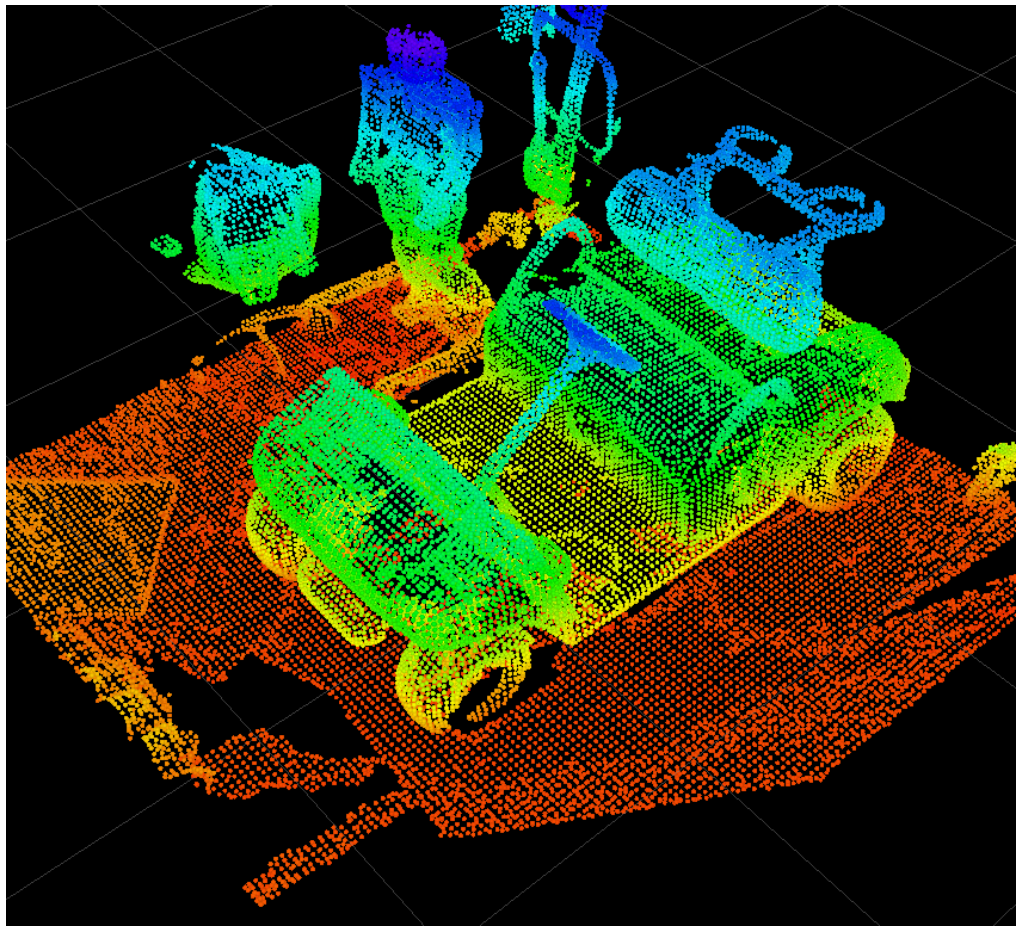
- 2 x Asus RGBD camera
- Tilting Hokuyo UTM-30LX-EW
- Long-range stereo cameras (Pt. Grey Flea3)
- Microstrain 3DM-GX3-45 IMU with GPS

TEPRA paper on "Functional Labeling of Utility Vehicle Point Clouds"

- Where is vehicle?
  - Map area around robot
  - Detect vehicle in map
- Given arbitrary sensor view, register to vehicle
  - Isolated static views here, but will be incrementally updated when running live
- Find functional vehicle parts within registered sensor view -> set goal to walk toward, grasp, etc.
  - Floor

Where is the Vehicle?

1. Make map. PCL KinFu mesh created from handheld Asus



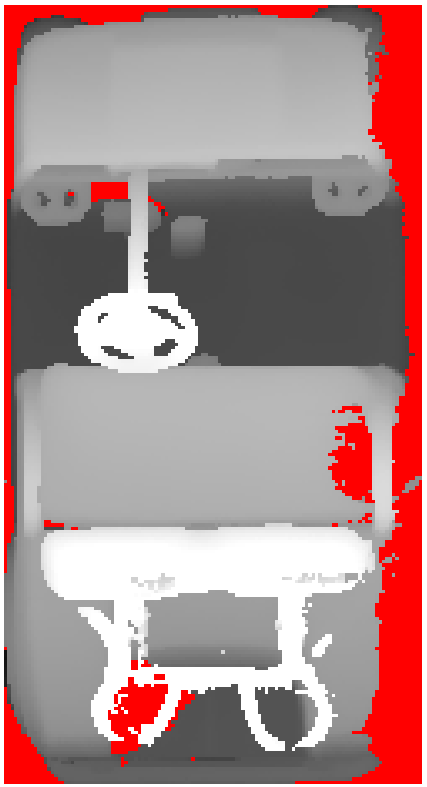
Where is the Vehicle?

1. Make map. PCL KinFu mesh created from handheld Asus
2. Find ground plane, convert to heightmap

Red pixels = ground, intensity proportional to height

Where is the Vehicle?

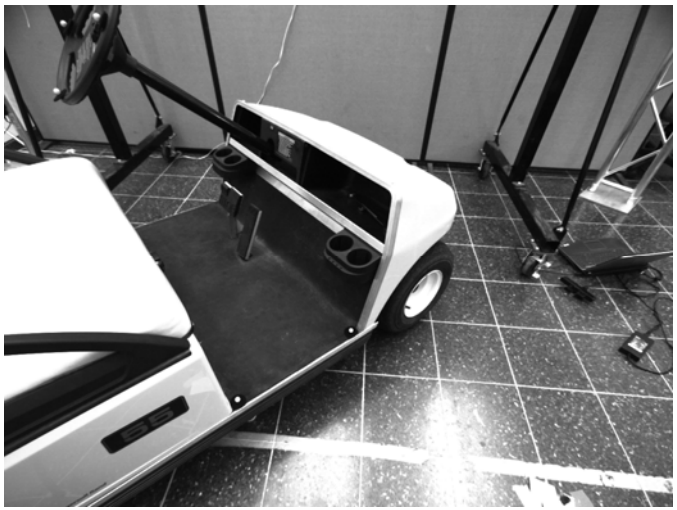
1. Make map. PCL KinFu mesh created from handheld Asus
2. Find ground plane, convert to heightmap
3. Search for vehicle-sized rectangle with surrounding height contrast

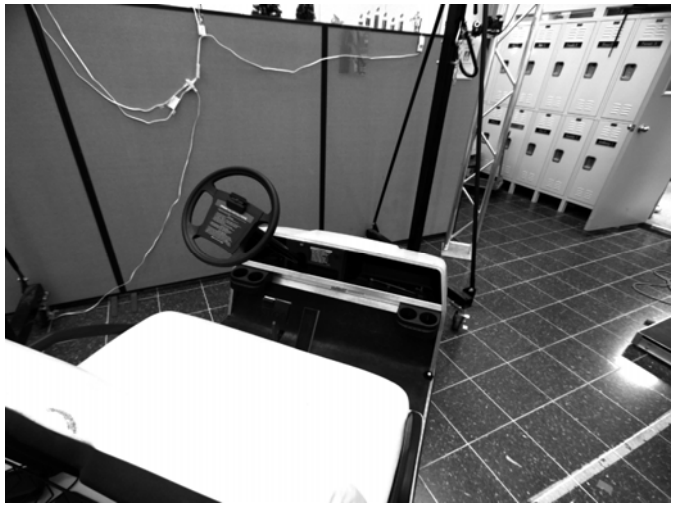


Learning Vehicle Geometry

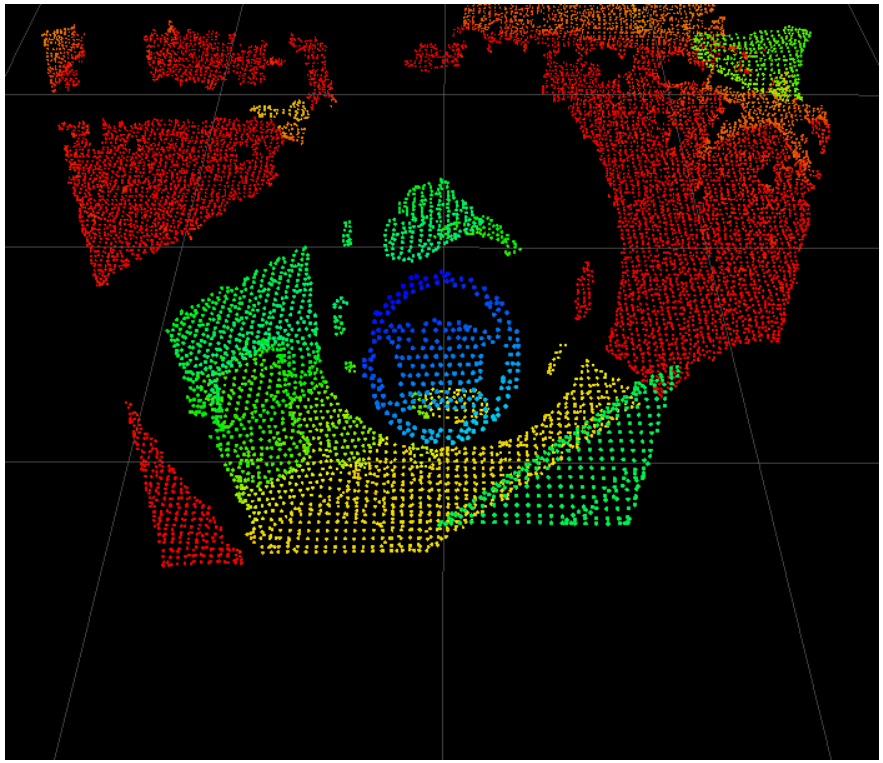
- Get ranges on length, width, aspect ratio, ground clearance from published specifications of 7 representative vehicles
  - Ground clearance + delta = ~floor height
- Steering wheel
  - Range of diameters from vendor websites
  - Tilt/pitch range anecdotal

Sensor View Registration, aka "What Part of the Vehicle is This?"

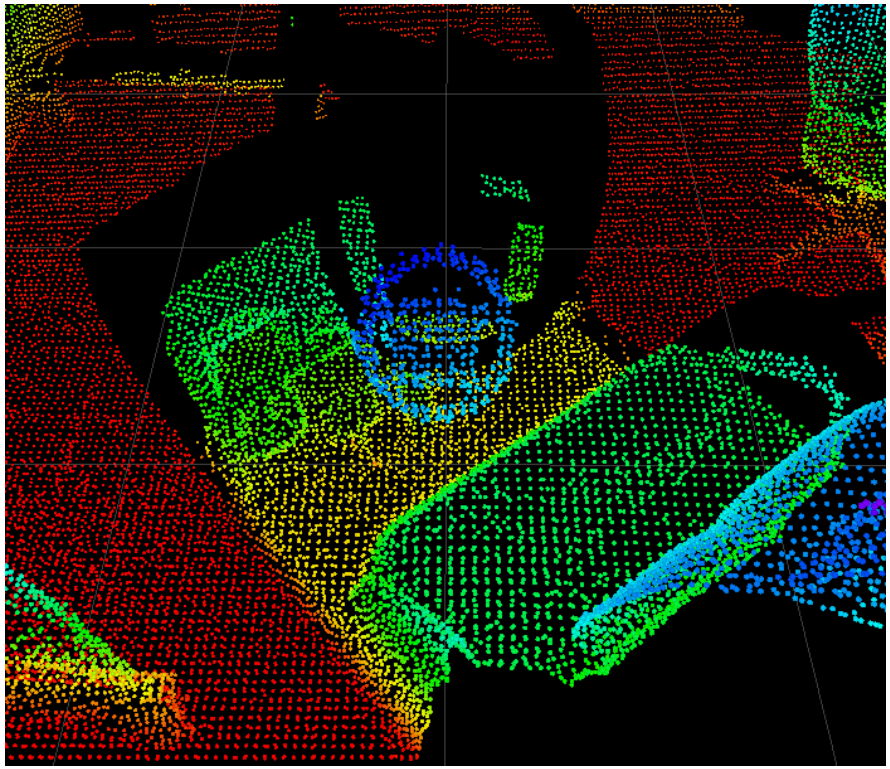




Sensor View Registration



Asus point cloud



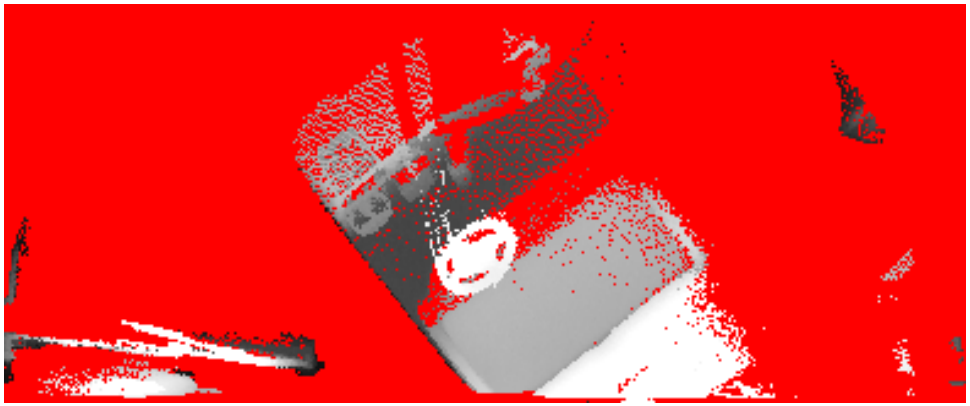
Hokuyo point cloud

Sensor View Registration

- Find ground plane, make *sensor heightmap*
- Now registration is 2-D image template matching problem: (dx, dy, dtheta)

Asus

Hokuyo



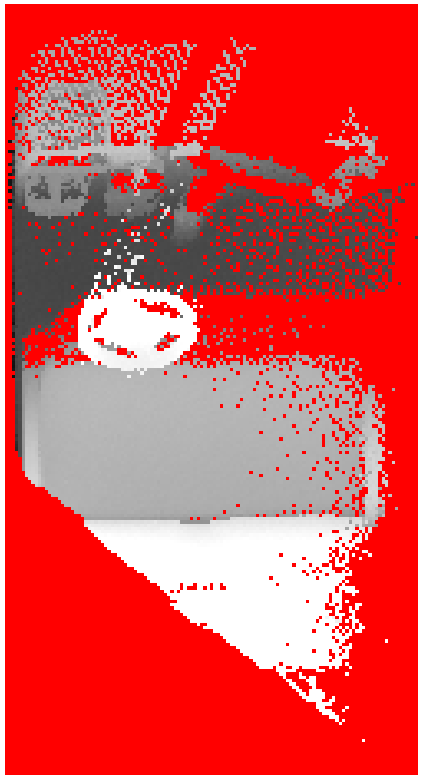
KinFu vehicle heightmap

Sensor View Registration

- Typical approach would be to find features, hypothesize matches, estimate transform, but missing data complicates standard OpenCV feature computation
- Formulate pixelwise objective function for hypothetical (dx, dy, dtheta) which counts height matches
- Multiscale exhaustive search takes ~1 minute
  - When we are tracking this should work in real time because (dx, dy, dtheta) will only change incrementally

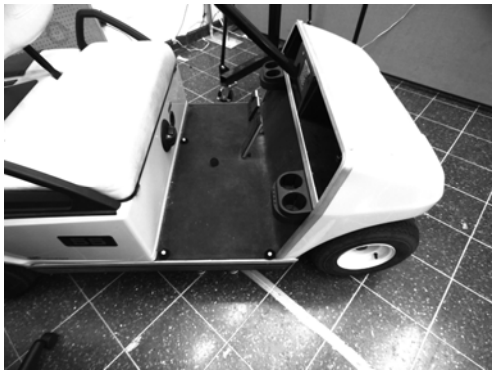
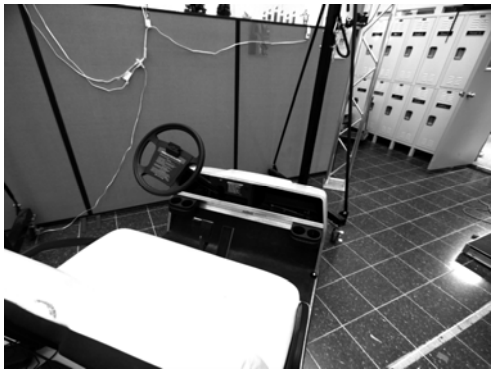
Sensor View Registration Results

- Trim away non-vehicle points
- Now we can search for functional parts in vehicle coordinates

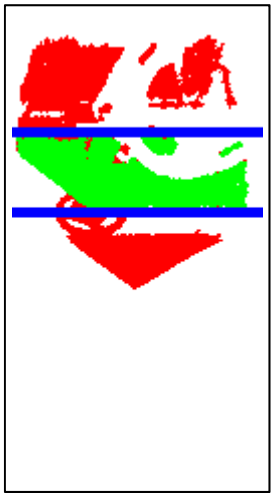
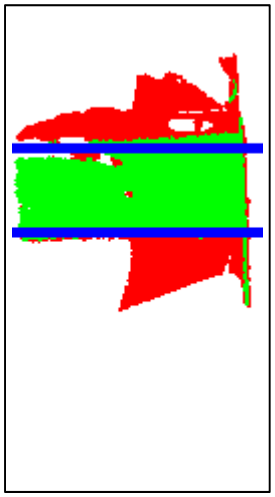
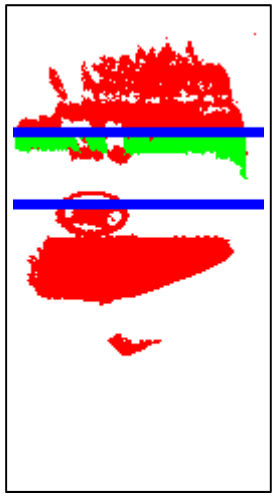




Finding Vehicle Parts: Floor

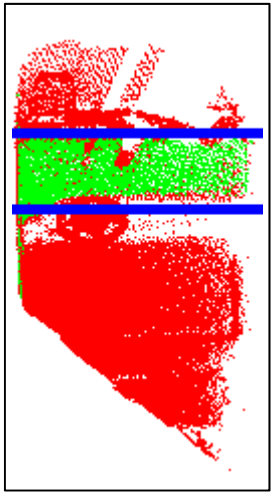
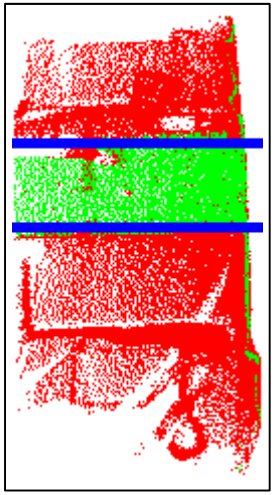
- What do we know?
  - Rectangle in plane (nearly) parallel to ground
  - Rectangle spans vehicle, aligned with its axes
  - Height within reasonable range
- Method (search space = [*x, length*])
  - Robustly fit floor plane after z slicing
  - Search for forward/backward limits x_{max}, x_{min} of rectangle which maximizes ($N_{inliers}$ - $N_{outliers}$) / area

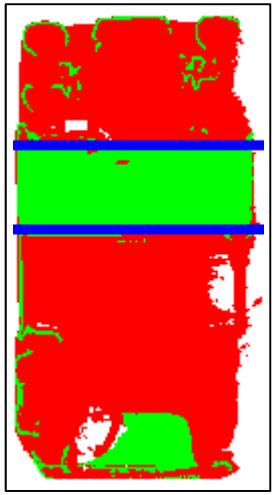
Asus

Hokuyo

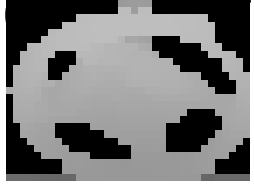
No floor plane found for Hokuyo

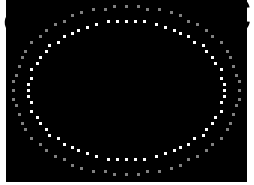


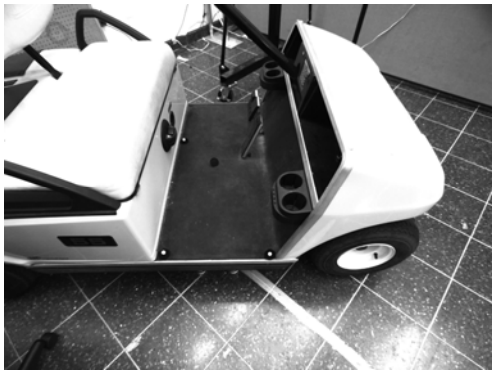
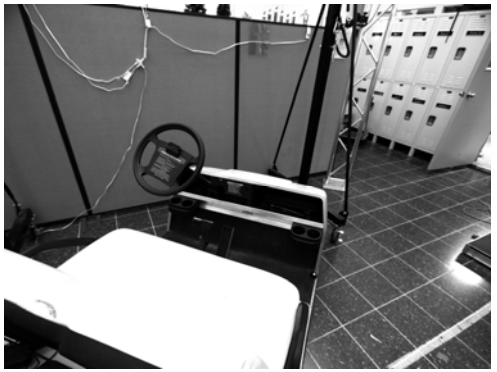
KinFu

Finding Vehicle Parts: Steering Wheel

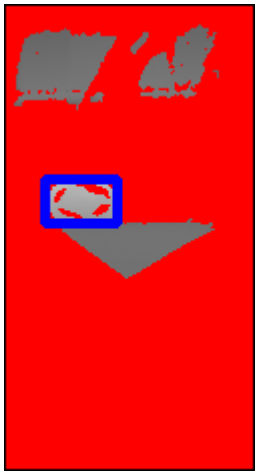
- What do we know?
  - Tilted circle within tight range of diameters -> ellipse in heightmap with axes aligned with vehicle
  - Rough height range
  - Inset from vehicle edges (don't actually know front from back, not assuming left-hand drive)
- Method (search space = [*x, y, phi, r*])
  - No plane fit--just directly hypothesize ellipse (derived from diameter, tilt), location in heightmap
  - Measure height contrast around circumference of ellipse

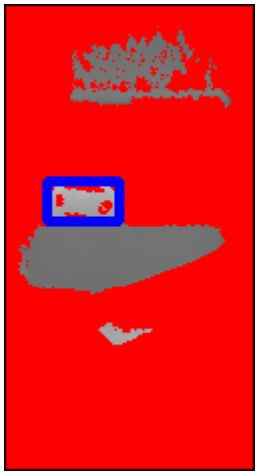




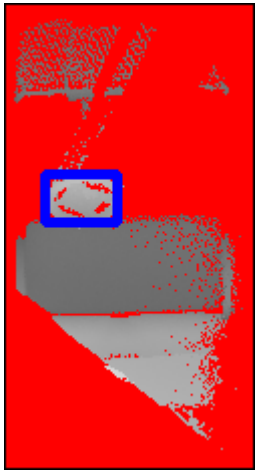




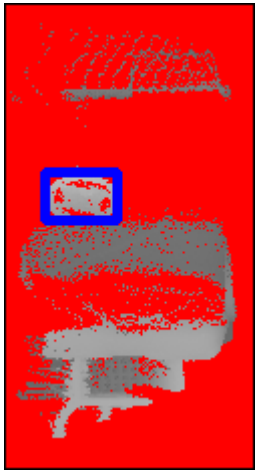
Asus

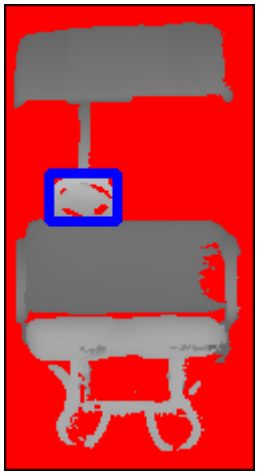




Hokuyo





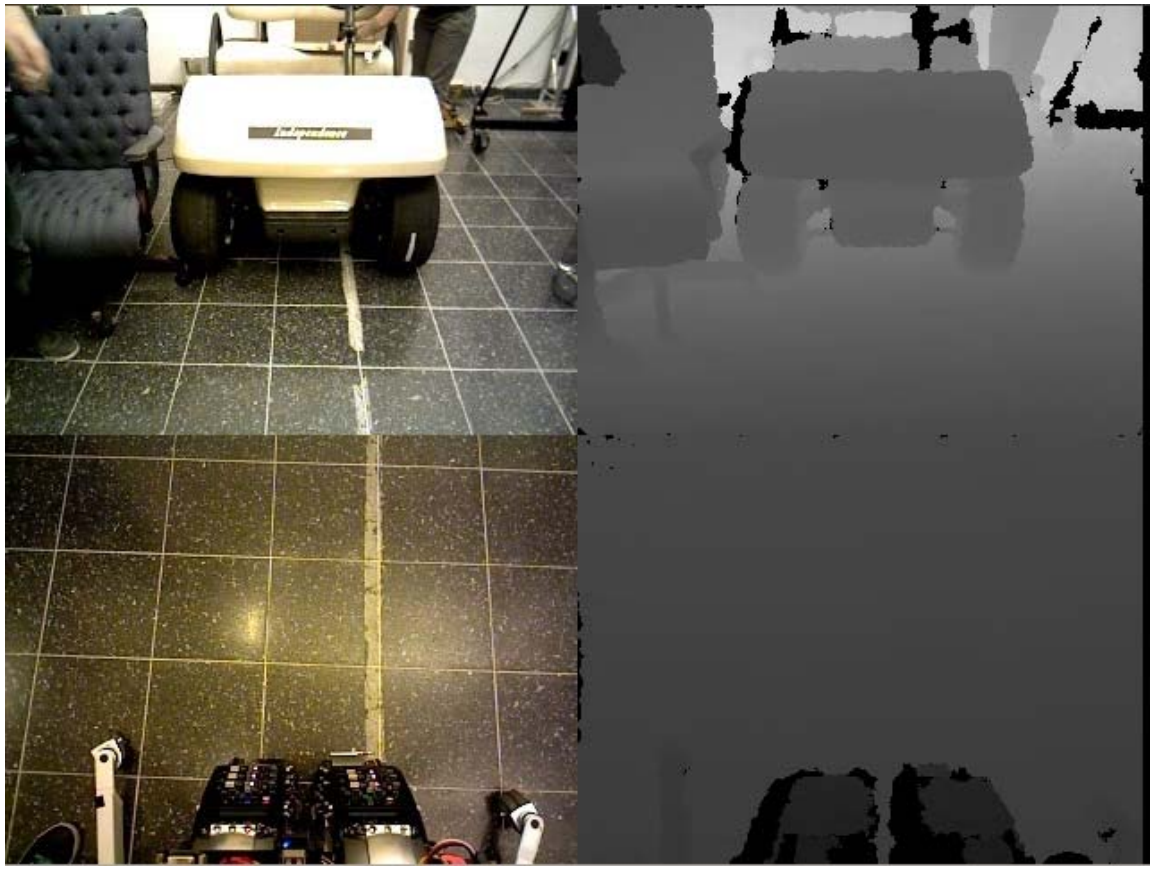


KinFu

Recent Work on Vehicle Approach

Recent Work on Vehicle Approach

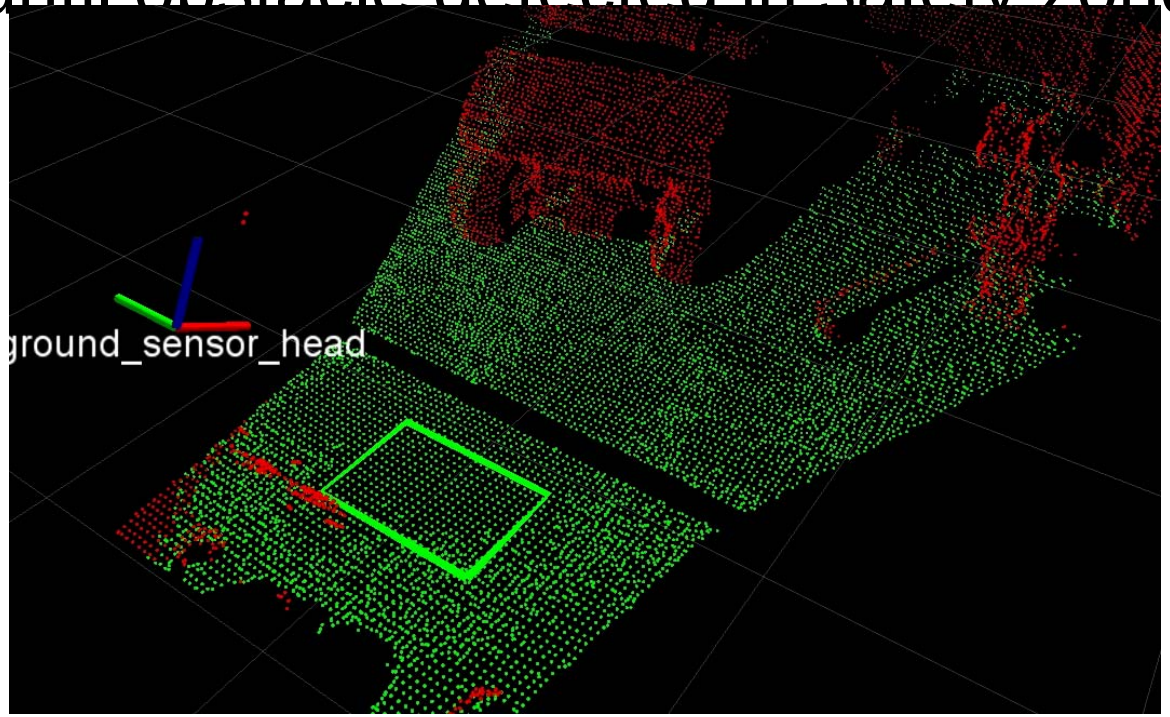
- Added downward-pointing Asus -> combined FOV of ~60 x ~90 degs



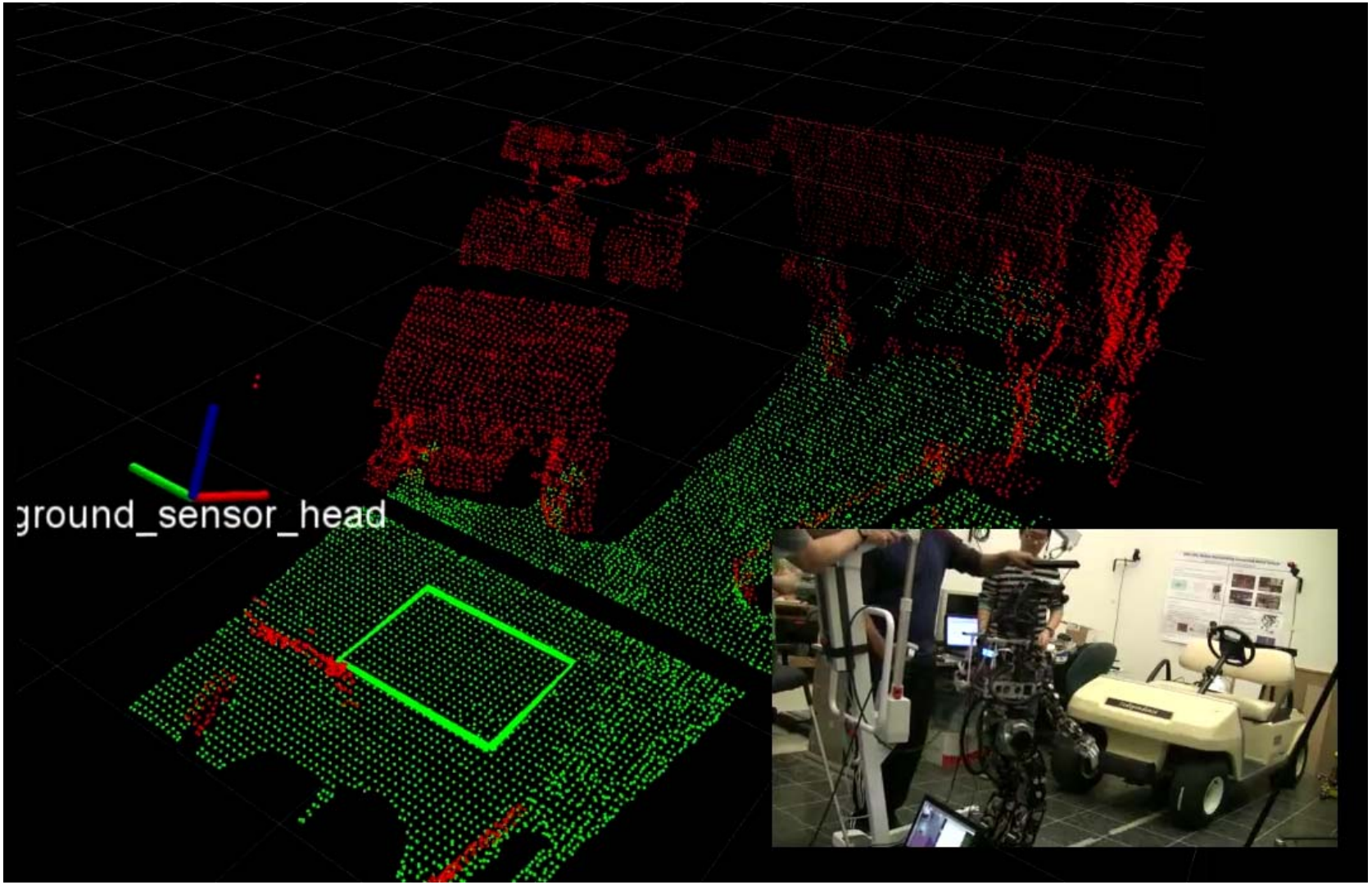
play movie

Recent Work on Vehicle Approach

- Use IMU for 3-axis orientation, rough robot height to initialize ground tracker
- RANSAC plane fit on decimated Asus point clouds
- Walk until obstacle detected in safety zone



Recent Work on Vehicle Approach



play movie

Next Steps

- Approach
  - Learn generic vehicle model, extract detailed geometric parameters from 3-D models of vehicles gathered with KinFu
  - Find vehicle before starting to walk
  - Plan path to passenger side and follow
  - Step-up: need to deal with possible roof, door
- Driving
  - Online calibration of steering, acceleration parameters after we start to move
  - Data collection from manually driven vehicle to validate obstacle detection, road following perception