Implementation of an Oriented Bounding Box Distance Sensor for Virtual Test Drive

Tiger Mou (Vanderbilt University)

Eike Möhlmann (OFFIS)

Background

- Lots of research in high level autonomous driving
- Huge potential in global market
- MUST be safe and reliable
- Needs to be thoroughly tested
- Accidents in live tests are a huge potential for bad reputation
  - Uber, Tesla, Google

Valet Parking

- Start small and scale up
- Work towards higher levels of autonomy

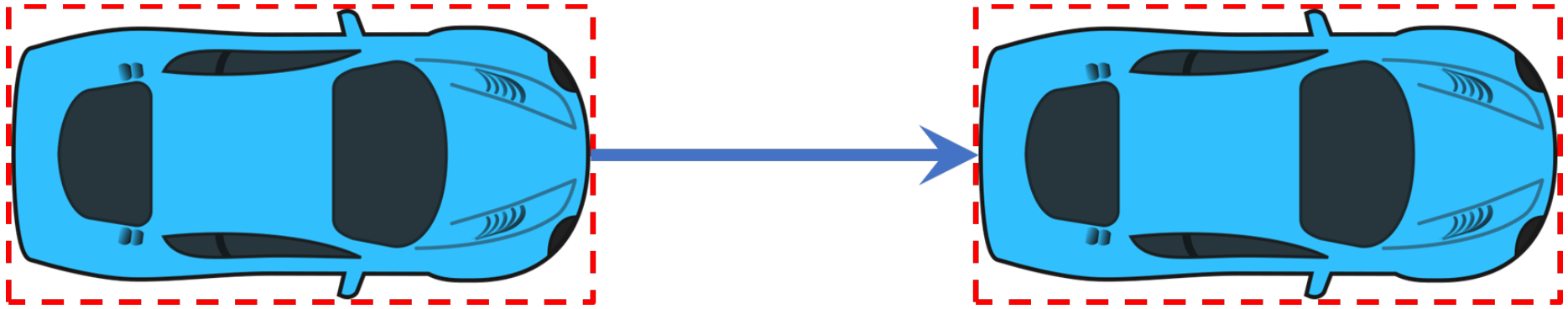
- Parking area management determines driving path
- Vehicle autonomously parks/returns

Virtual Test Drive (VTD) by VIRES

- Feature rich vehicle simulation platform
- Computer generatable scenarios
- Automated testing/verification capable
- Has room for additional features and plugins

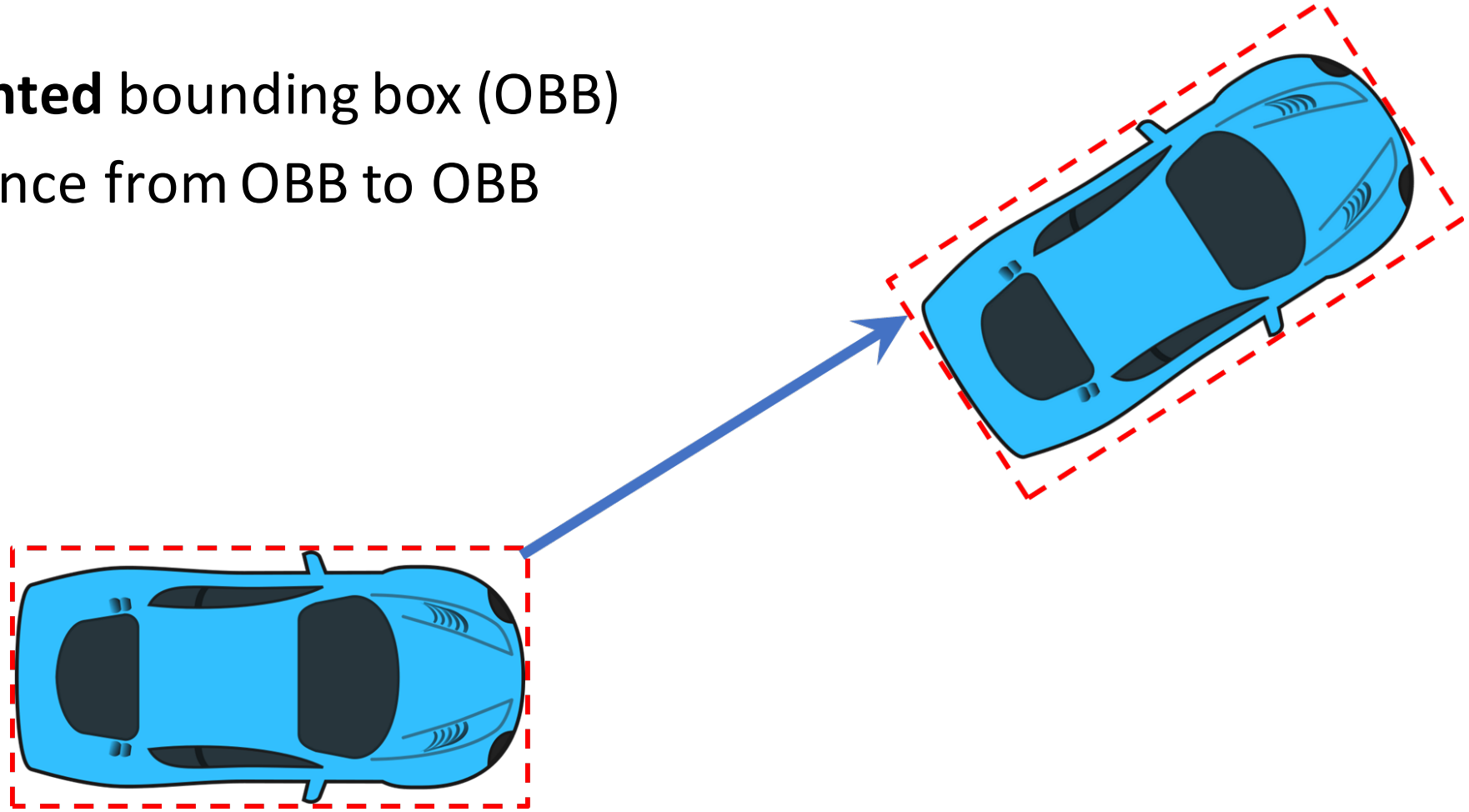
Oriented Bounding Box Distance Sensor

- **Oriented** bounding box (OBB)
- Distance from OBB to OBB



Oriented Bounding Box Distance Sensor

- **Oriented** bounding box (OBB)
- Distance from OBB to OBB



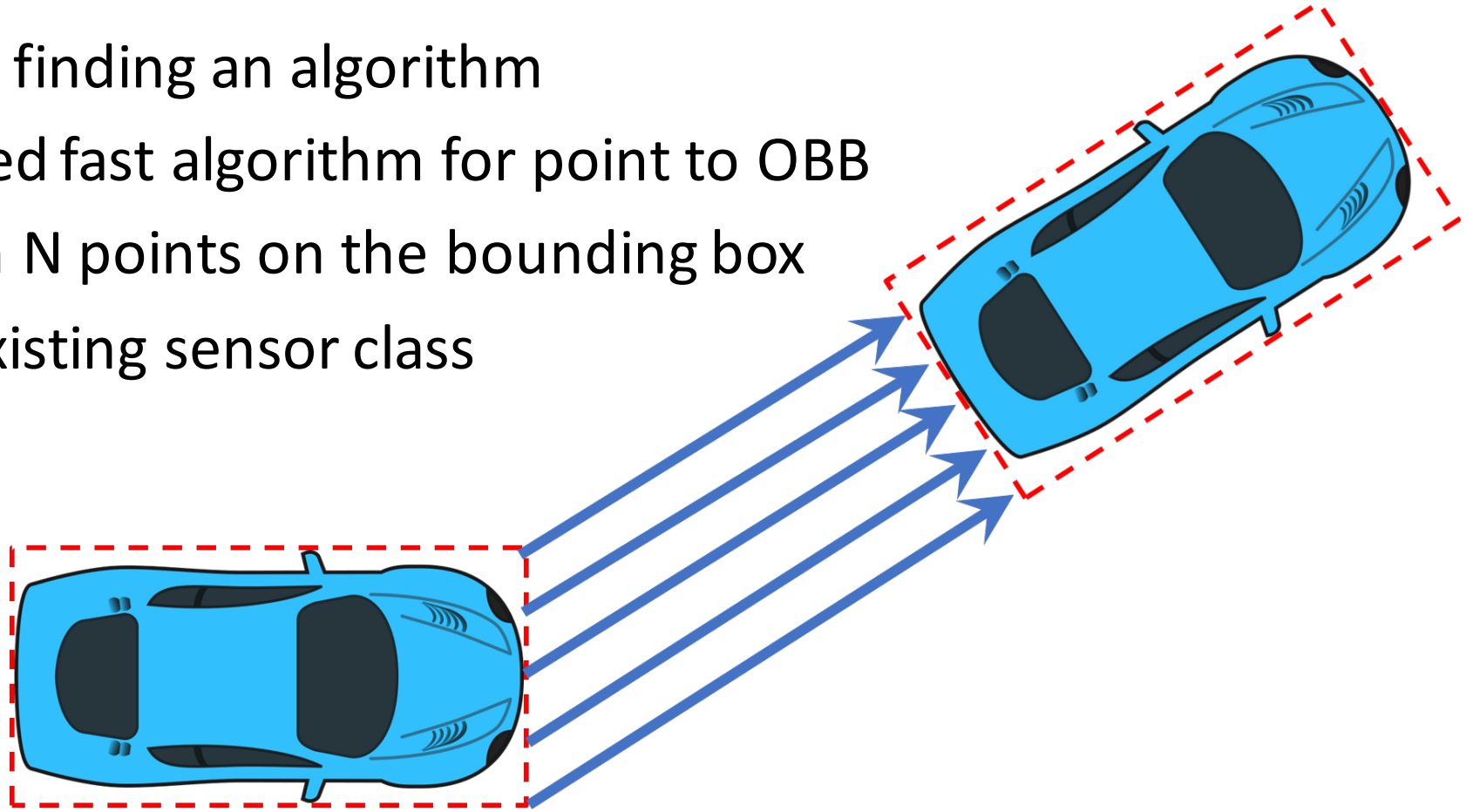
Oriented Bounding Box Distance Sensor

- VIRES likely has a distance sensor, but we need more
- Could use an external listener but not good enough

- Our own sensor would be customizable
- Lets us easily perform complex computations from **within** the simulation
- Send only what we need over the runtime data bus (RDB)

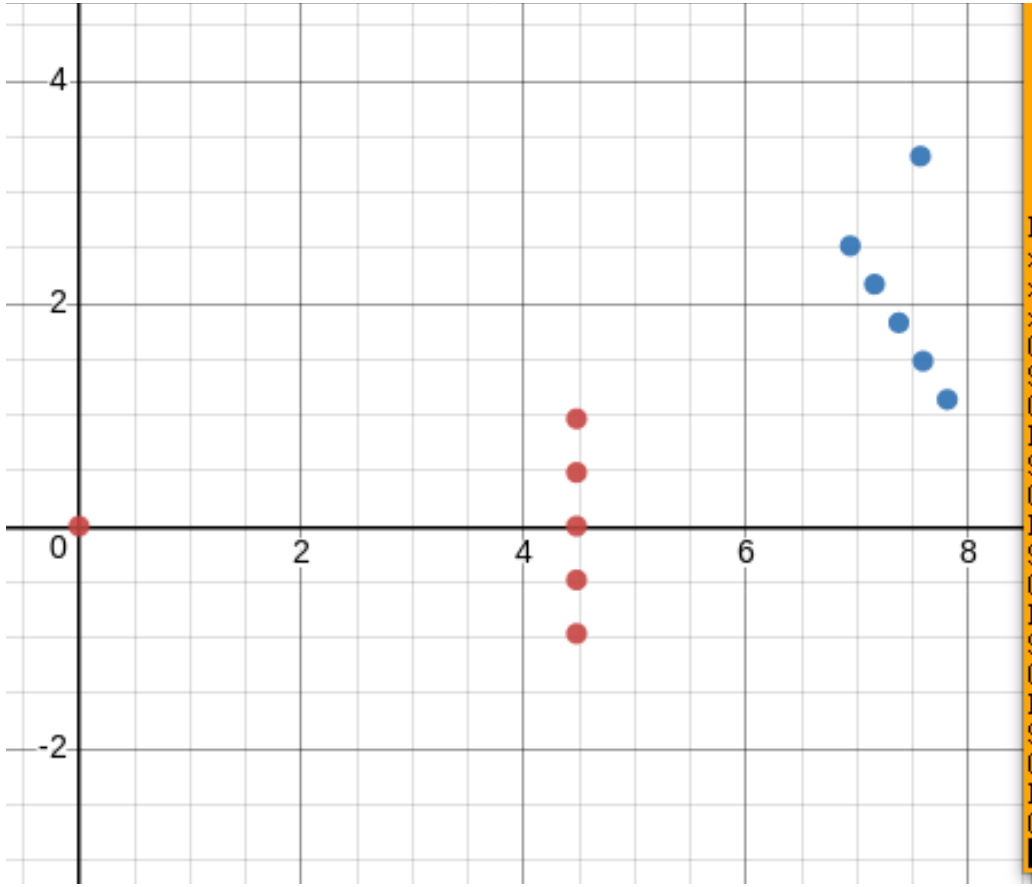
My Approach

- Had difficulty finding an algorithm
- VIRES provided fast algorithm for point to OBB
- Estimate with N points on the bounding box
- Extends an existing sensor class

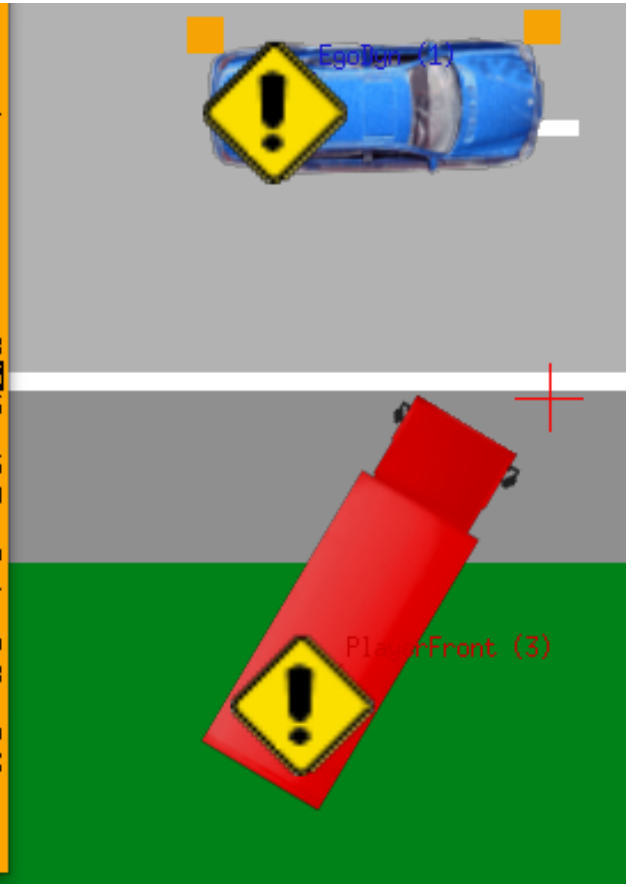


Sensor Testing

- Start small – only points on the front

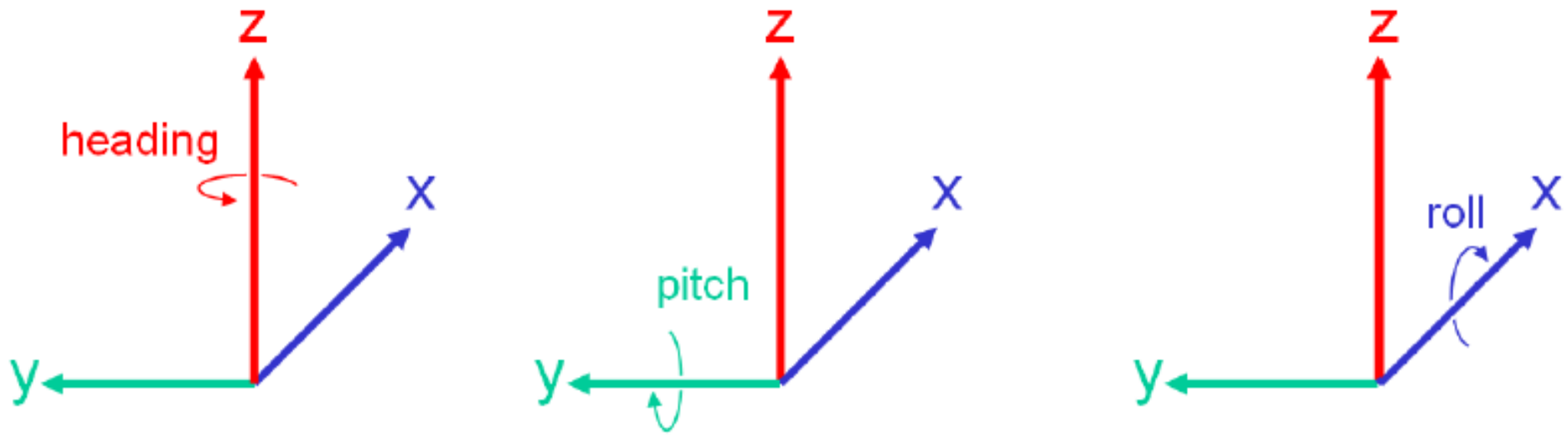



```
  POS:
    x/y/z = 6.830253 / 4.502139 / -0.000023
    h/p/r = -1.007840 / -0.000004 / 0.001125
    flags = 0x3, type = RDB_COORD_TYPE_USK, syst
  GEOM:
    dim x / y / z = 4.646 / 1.926 / 1.673
    off x / y / z = 1.390 / 0.000 / 0.000
BB Axes:
x: 0.533688  y: -0.845681  z: -0.000954
x: 0.845681  y: 0.533688  z: 0.000600
x: 0.000004  y: -0.001125  z: 0.999999
Calculated pos: x: 7.571282  y: 3.326143  z: 0.8
SENSOR POS: x: 4.477000  y: -0.966000  z: 1.2705
CLOSEST PT: x: 7.812522  y: 1.138960  z: 1.27285
DIST: 3.94418
SENSOR POS: x: 4.477000  y: -0.483000  z: 1.2705
CLOSEST PT: x: 7.594530  y: 1.484391  z: 1.27269
DIST: 3.68641
SENSOR POS: x: 4.477000  y: 0.000000  z: 1.27050
CLOSEST PT: x: 7.376538  y: 1.829822  z: 1.27254
DIST: 3.42864
SENSOR POS: x: 4.477000  y: 0.483000  z: 1.27050
CLOSEST PT: x: 7.158545  y: 2.175252  z: 1.27238
DIST: 3.17087
SENSOR POS: x: 4.477000  y: 0.966000  z: 1.27050
CLOSEST PT: x: 6.940553  y: 2.520683  z: 1.27223
DIST: 2.9131
Object is closest 2.9131 away.
```



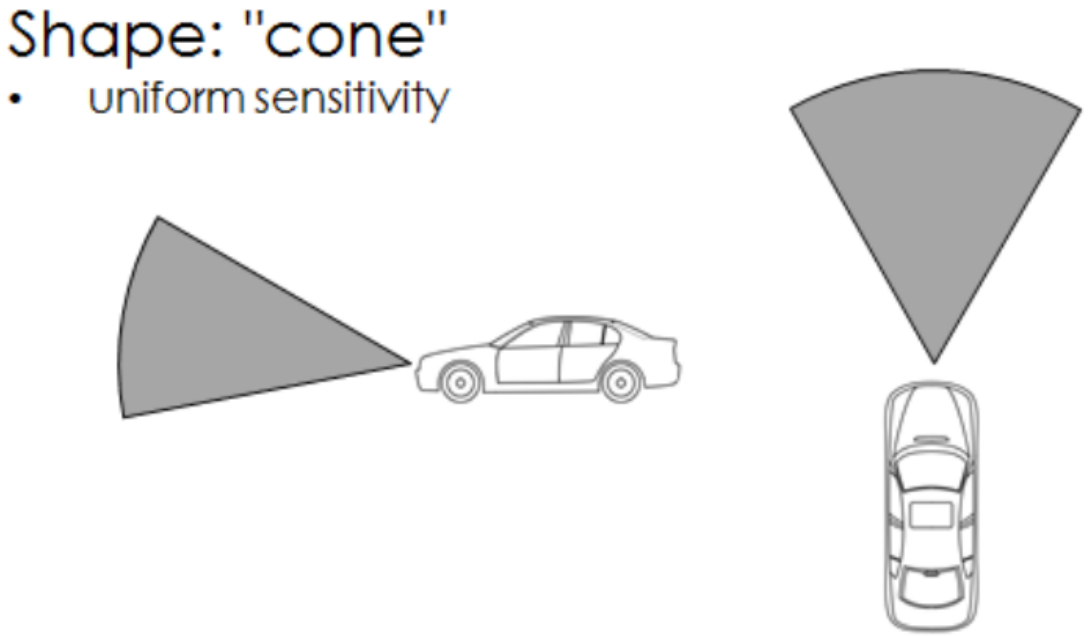
Additional Features

- “See Everything” functionality
- Ran into trouble with different coordinate systems



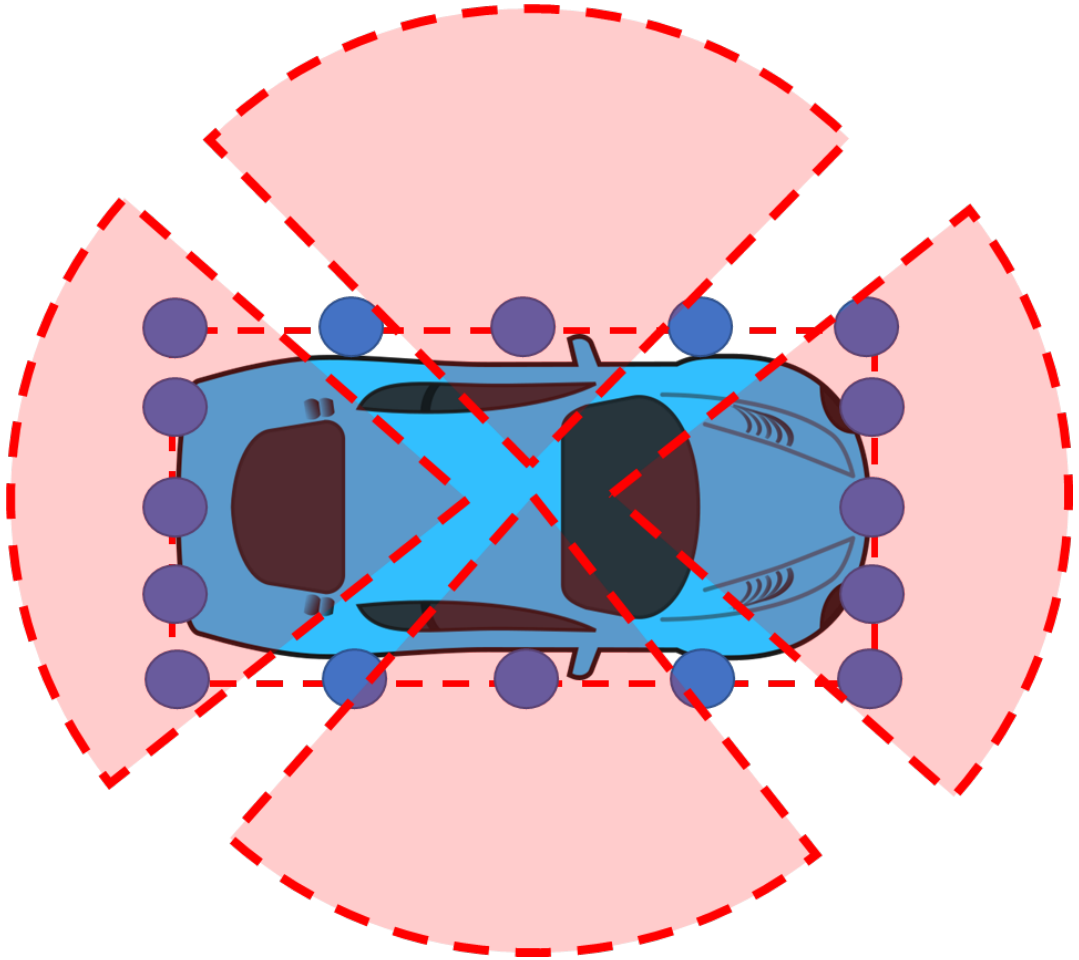
Additional Features

- Rotate the frustum with the wheels
- Can be useful for narrow frustums
- Potential downside: rotated frustum set in next frame, not current



Additional Features

- Add a sensor on each side
- Many possible approaches
- More calculation points
- Flag to enable/disable each side



Additional Features

- Filter the list of detected objects
- Send through the RDB the N closest or most critical objects
- Toggle with flags
- Helps focus on what is most important

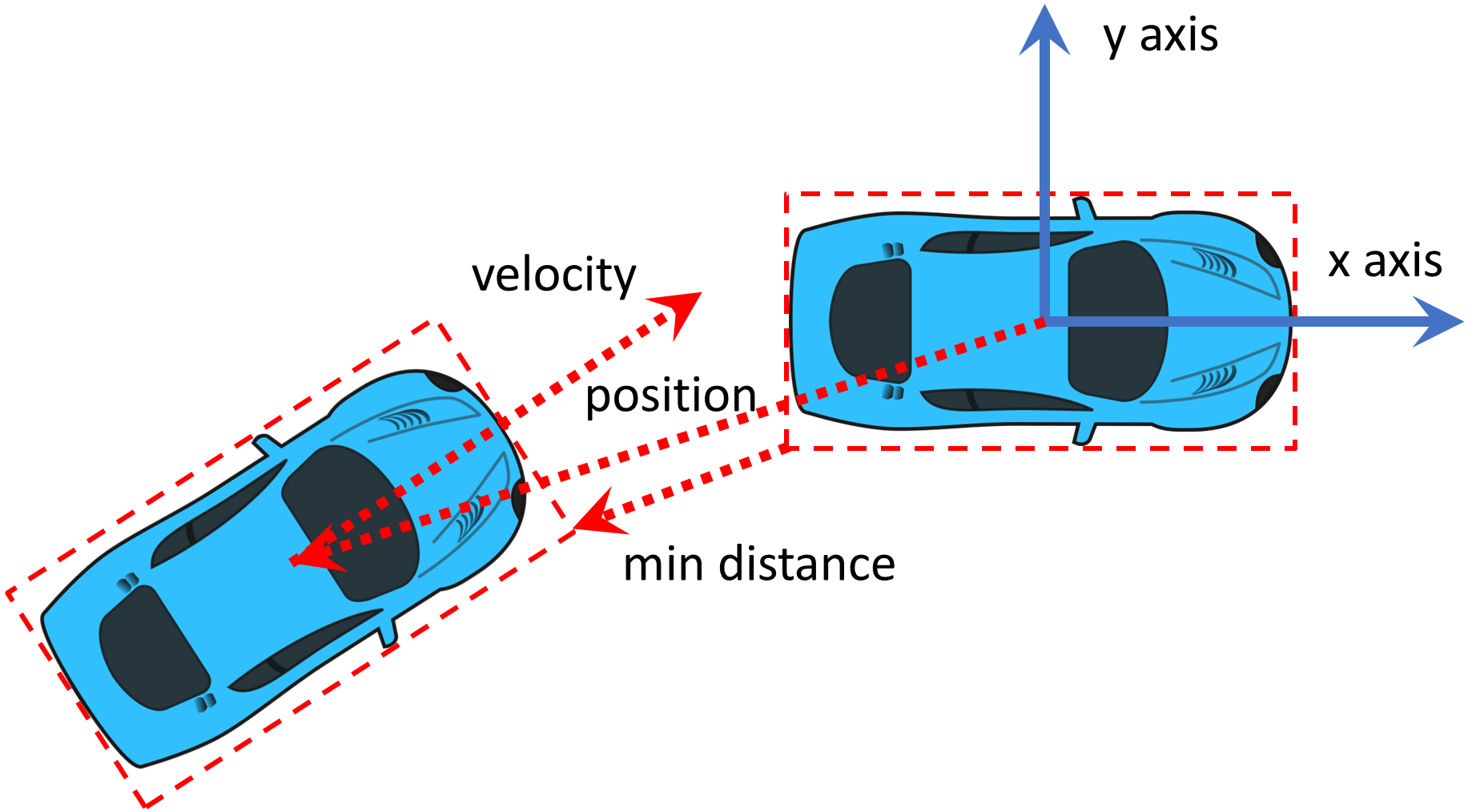
Additional Features

- Send distance data through the
  - **sensor's** RDB port
  - **main** RDB port (Unsuccessful)
- Each with flag to enable/disable

Basic Criticality Calculation

- Implemented basic criticality function
  - Help set up code structure
  - Work out some potential bugs
- Sensor coordinate system
  - Positions and velocities relative to sensor
  - Dot product of unit vectors position and velocity determine if object is approaching or not
- Velocity vector towards origin = collision path
  - Use previously calculated distance to estimate time to collision

Basic Criticality Calculation



Valet Parking Scenarios

- Use the Scenario Editor to create a few simple valet parking scenarios

Future Work

- Improve distance algorithm, add tests
- Improve coordinate system flexibility
- Improve/fix coordinate system conversion algorithm(s)
- Improve criticality calculation
- Add more information to the RDB structs
- Better filtering
- Fix send through main RDB port
- Other necessary features
- Valet parking scenarios