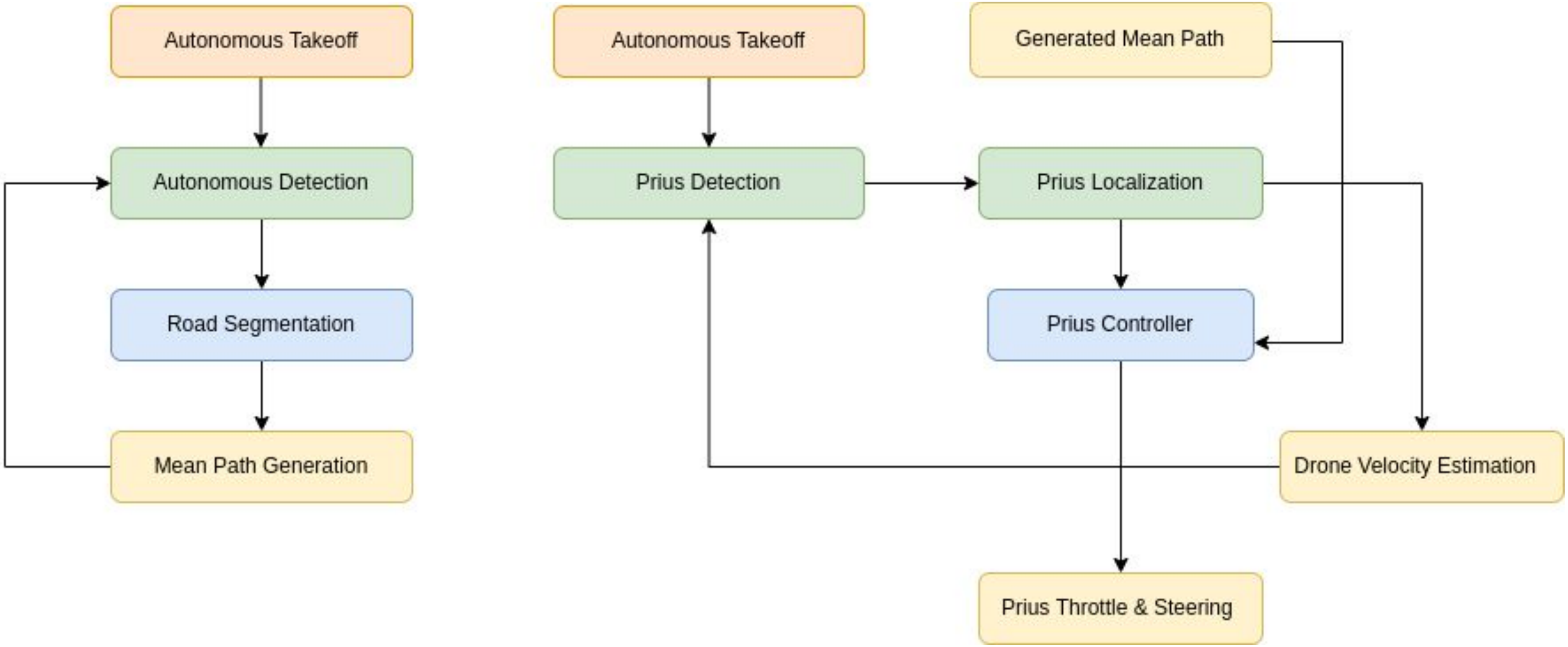
D.R.D.O UAV Guided UGV Navigation Challenge

TEAM 10

INTER IIT TECH-MEET 10.0

Approach



Autonomous Takeoff
Autonomous Detection
Road Segmentation
Mean Path Generation
Autonomous Takeoff
Prius Detection
Generated Mean Path
Prius Localization
Prius Controller
Drone Velocity Estimation
Prius Throttle & Steering

Approach (contd.)

SEGMENTATION

Road is segmented from the pointcloud using RANSAC along with some heuristics and algorithms for eliminating rogue data.

MEAN PATH DETECTION

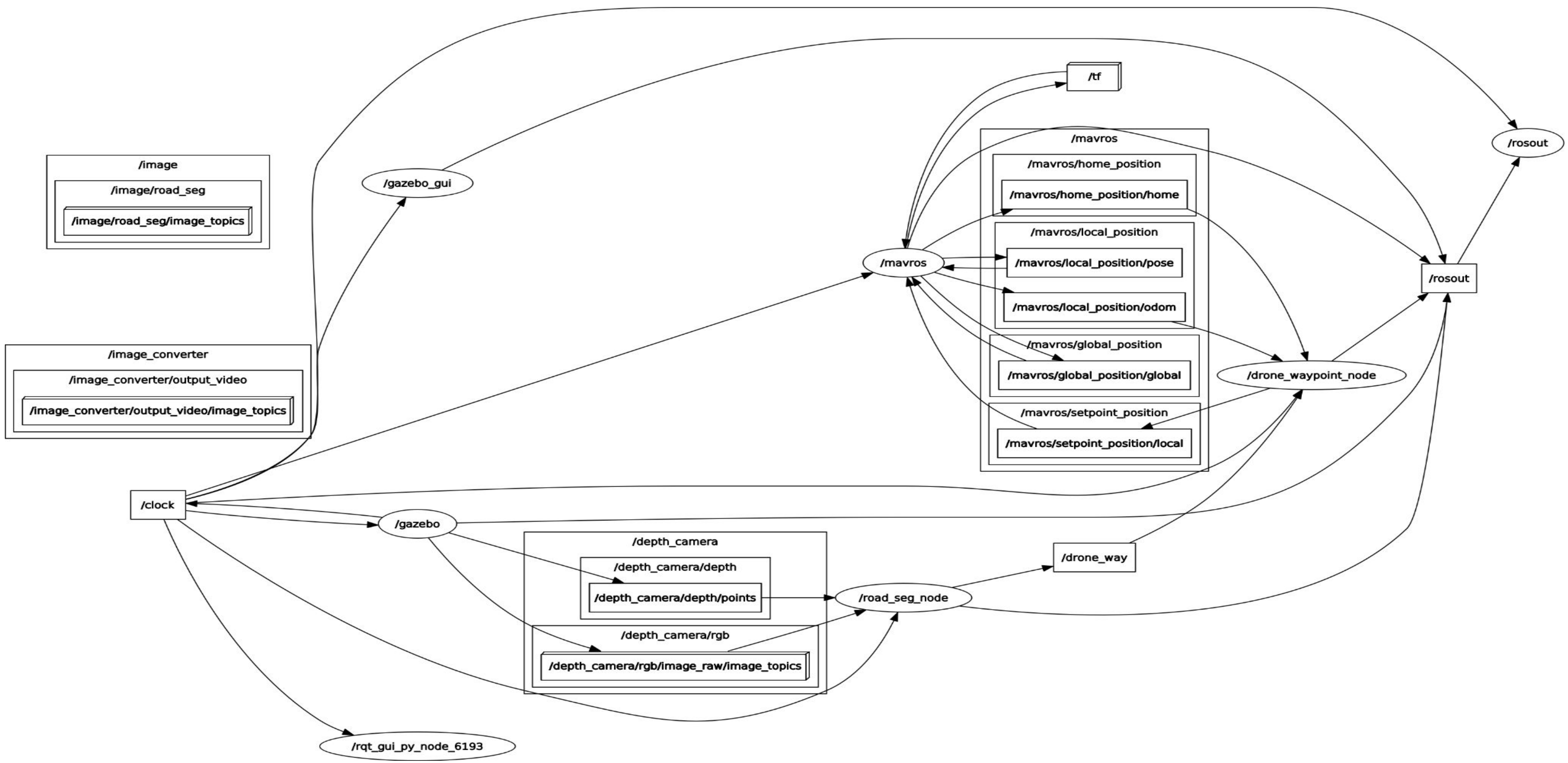
MinRectArea and PCA module of OpenCV library for waypoint calculation

Frame transformations to get the waypoints in map(global) frame

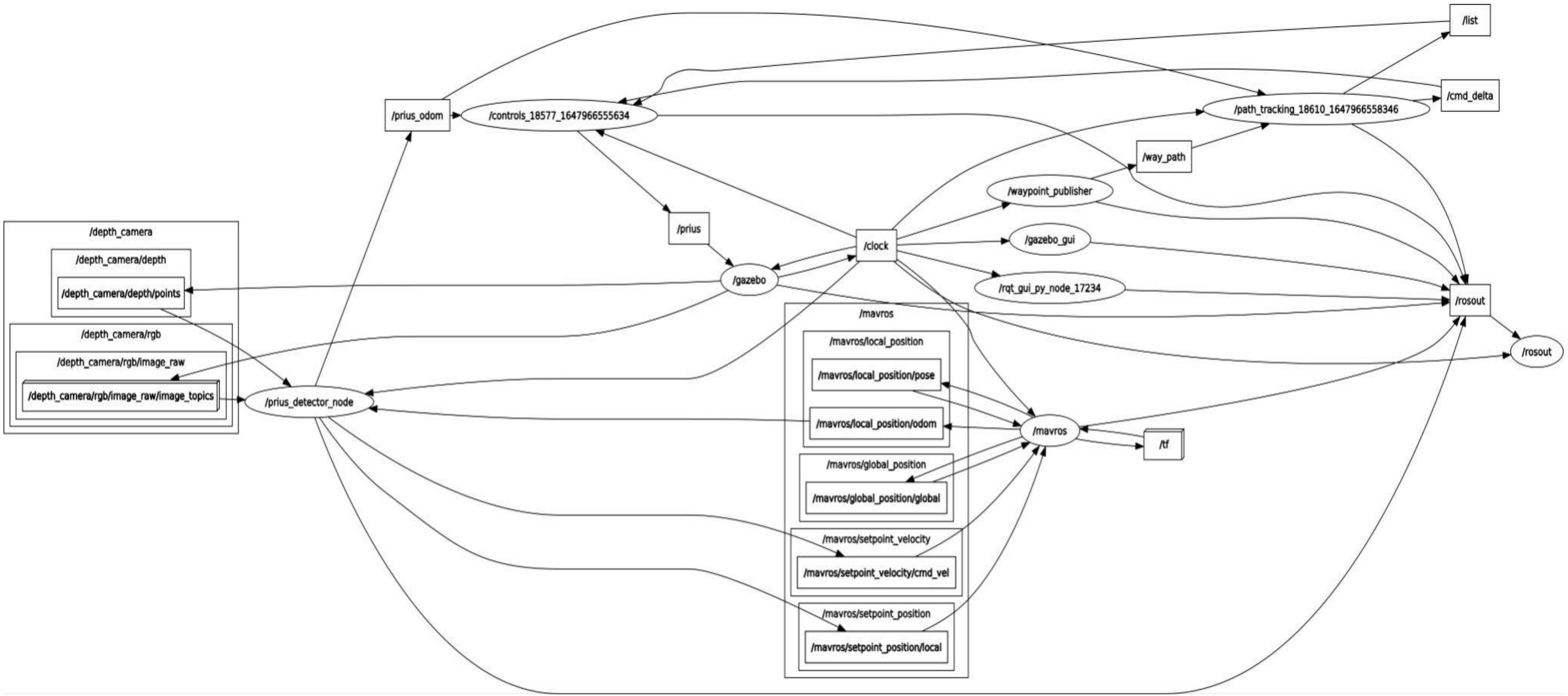
CONTROLLER

The prius is navigated along produced waypoint using controller. The UAV camera is used provide visual feedback for minimal navigation error.

Software Architecture Graph: Segmentation



Software Architecture Graph : UAV Guided UGV



Software Architecture: Mapping Nodes

road_seg_node

Segment the road from the pointcloud obtained from camera and find mean path to calculate further waypoints for the drone and prius.

mapping_fsm_node

Converts the waypoints from camera frame to drone frame and then further to world frame.

Software Architecture: Controller Nodes

Prius Controller

It is responsible for waypoint publishing, steering and throttle/brake control of the UGV

- Pure Pursuit Controller
- Path Publisher
- PID velocity controller

Prius Detection & Localization

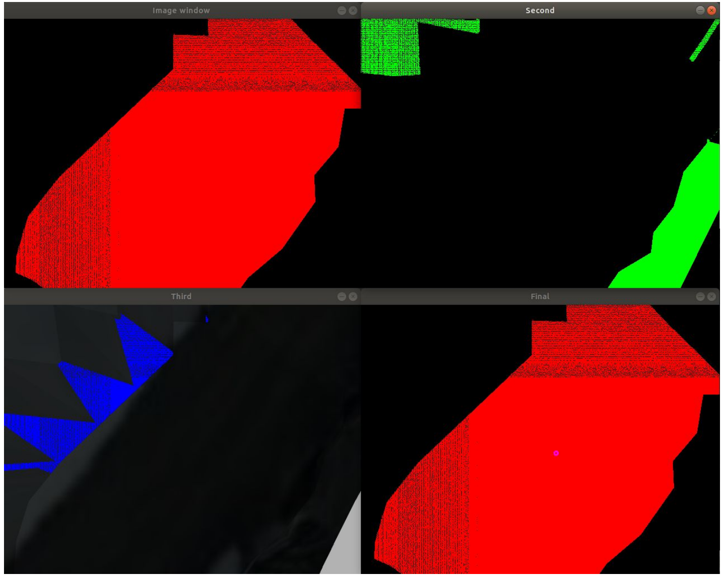
Detect prius and find its position, orientation and velocity in world frame.

Road segmentation

- pcl::RANSAC algorithm is used to get three planes from pointcloud
- Z-value (average slope) computed for planes and small size penalty applied, if applicable. Planes are sorted on this basis.

Case 1: Ranking is conclusive

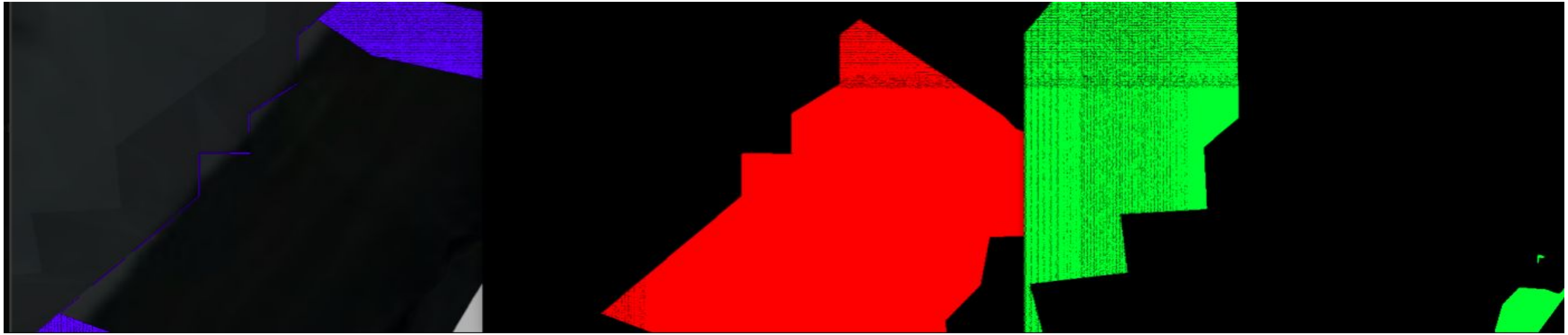
- Plane with lowest Z-value is selected as road. *prevZ* value is updated.
- prevZ = prevZ *(1 - gamma) + average_depth_of_road * gamma



Road segmentation(contd.)

Case 2: Ranking is uncertain and inconclusive

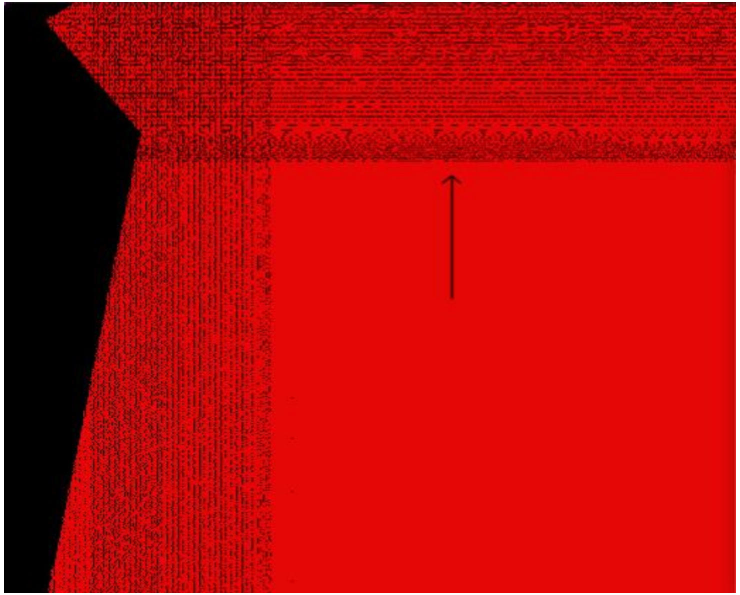
- Time based filtering is applied using the prevZ value.
- Average depth is computed for planes with very close Z-values.
- The plane with the closest depth to the prevZ value is selected as road.
- prevZ value is updated.



Mean Path Detection

The module uses three avenues of approach in concert with each other.

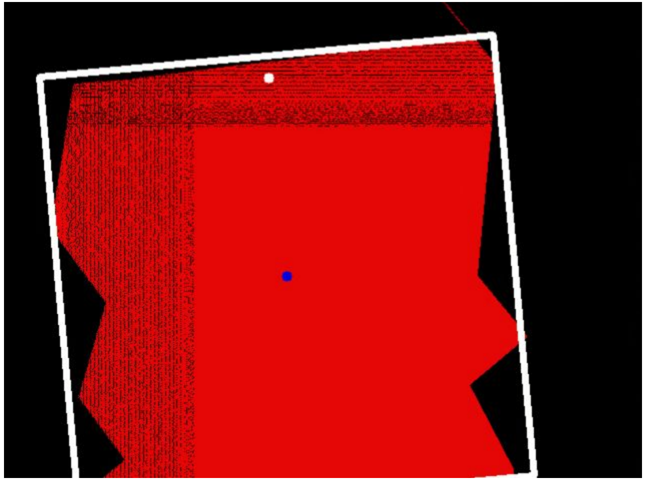
- **Situation 1: When major part of the road is detected**
  - Calculated ratio of pixels of road(N_0) by total number of pixels(N) and compared it to certain threshold.
  - In this situation drone was given waypoint to move straight on the road.

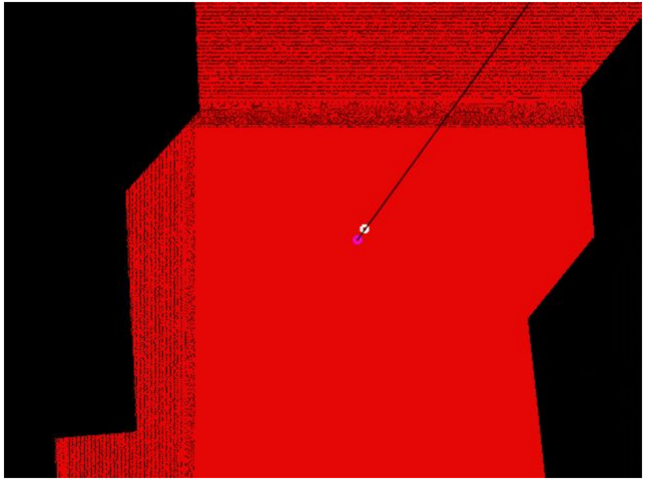


Mean Path Detection(contd.)

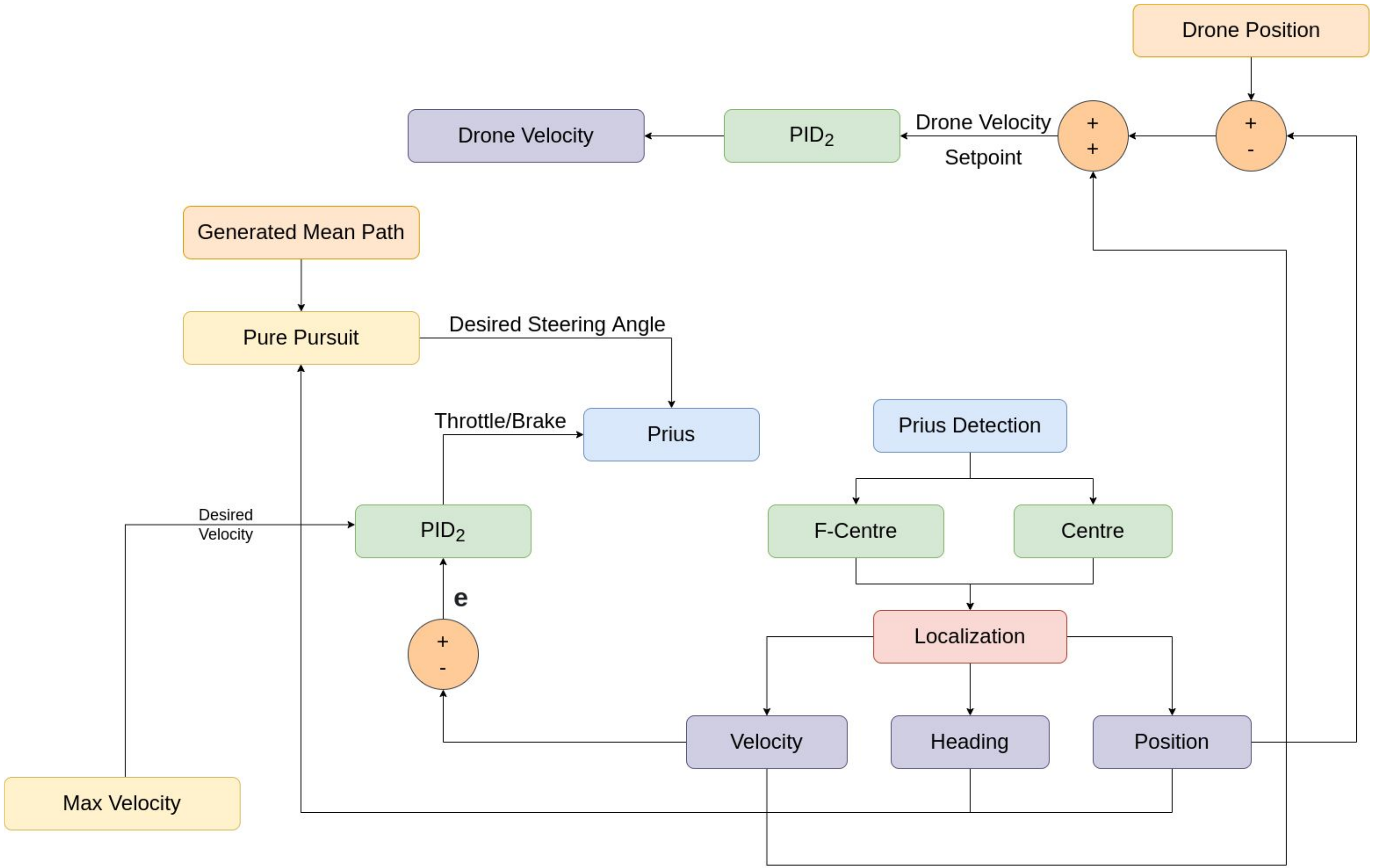
- **Situation 2: When only a part of road is detected.**
  - cv::MinAreaRect was used to fit a rotated rectangle around road.
  - If ratio of area of Rectangle(A_R) and Area of Contour of road(A_c) is smaller than a given threshold , drone was given waypoint in the direction of orientation of road.
  - Otherwise,cv::PCA was used to get direction of movement.

Corrective measure -If the detected road center was too far from image center, drone was directed to rectify the offset. Waypoint generated was not used for prius.





UAV Guided UGV Algorithm

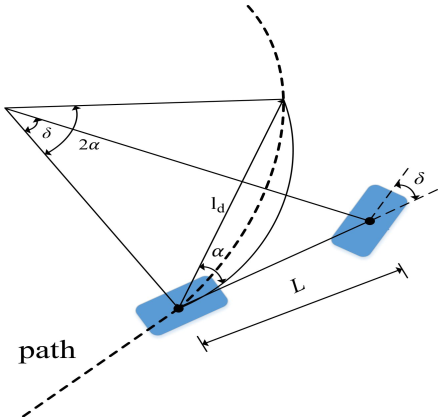


UGV Path Follower guided by UAV

➔ UGV Controller for Path Following

- <u>Pure Pursuit Controller</u>:
  Returns the required steering angle to follow the desired path by considering the following parameters:
  - current position
  - heading angle
  - lookahead distance

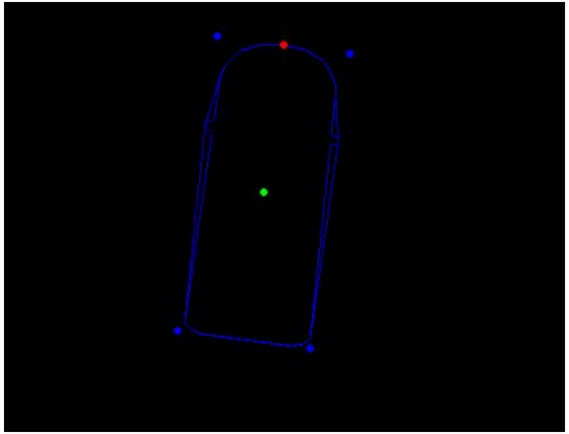
- <u>PID Velocity Controller:</u>
  Maintains the desired speed by giving required throttle/brake, considering the current speed.



➔ UGV Detection

Since UGV lacks any sensor we need to localize UGV. Following algorithms were used to detect the Prius.

- Inrange colour selection
- Gaussian Blur
- Convex Hull
- Feature Count (for robustness)
- Corner extraction via Rectangle fitting



Using the four corners of prius, we can find the center point of the prius and the midpoint between the front wheels.

➔ UGV Localization

- Coordinate transformation is performed on the required coordinates
- Data passed to control system in order to guide UGV with minimal error

Camera Frame → Quad Frame → World Frame → Control System

Prius velocity is calculated for feedback using change in center coordinates with respect to time.

$$\text{detected\_speed} = \left| d(x,y)/dt \right|$$

➔ UGV Tracking

- The UAV is given a velocity setpoint such that the UGV is always present in the camera's FOV.
- For calculating setpoint, UGV's pose, orientation and velocity is calculated using OpenCV algorithms.

drone_velocity(x,y) = k(prius_position(x,y) - drone_position(x,y)) + prius_velocity(x,y)

drone_velocity(z) = k_z(prius_position(z) - drone_position(z) + 18)

drone_angular_velocity(z) = k(prius_yaw - drone_yaw) + prius_angular_velocity(z)

Performance Analysis

Computation	**Cost**
Gazebo	~1.0 cores
Segmentation and Mapping	~1.0 cores
UAV guided UGV	~2.0 cores

Local System Specifications	
OS	Ubuntu 18.04
Processor	Intel® Core™ i7-10750H CPU @ 2.60GHz × 12
Graphics	GeForce GTX 1660 Ti/PCIe/SSE2

Previously tried approaches

- Normal Extraction for plane segmentation :- Relies on utilising neighbourhood data for a point which proved too computationally expensive.
- At a constant velocity UGV fails to take sharp turns. Tried to slow down UGV only at turns without affecting speed elsewhere. But drone fails to keep track at high speed.

Scope of improvement

- Add exploration routine to enable the drone into work even when the road leaves the frame or it encounters situations where it is unable to detect the correct direction.
- Making the Mean Path detection more robust even when the road segmentation module gives out suboptimal results.