Sensor Fusion and Navigation for Autonomous Systems Using MATLAB & Simulink

Abhishek Tiwari
Application Engineering
Smart autonomous package delivery

① Autonomous Driving

② Warehouse Automation

③ Last Mile Delivery

Manufacturer

Consumer
Capabilities of an Autonomous System

Some common Perception tasks:

- Design localization algorithms
- Design environment mapping algorithms
- Design SLAM algorithms
- Design fusion and tracking algorithms
- Label sensor data
- Design deep learning networks
- Design radar algorithms
- Design vision algorithms
- Design lidar algorithms
- Generate C/C++ code
Capabilities of an Autonomous System

- **Perception**
- **Planning**

Some common planning tasks:

- Visualize street maps
- Connect to HERE HD Live Map
- Design local and global path planners
- Design vehicle motion behavior planners
- Design trajectory generation algorithms
- Generate C/C++ code
Capabilities of an Autonomous System

Some common control tasks

- Connect to recorded and live CAN data
- Design reinforcement learning networks
- Model vehicle dynamics
- Automate regression testing
- Prototype on real-time hardware
- Design path tracking controllers
- Design model-predictive controllers
- Generate production C/C++ code
- Generate AUTOSAR code
- Certify for ISO26262
In this talk, you will learn

Reference workflow for autonomous navigation systems development

MATLAB and Simulink capabilities to design, simulate, test, deploy algorithms for sensor fusion and navigation

- Perception algorithm design
- Fuse sensor data to maintain situational awareness
- Mapping and Localization
- Path planning and path following control
Sensor outputs vary along the processing chain

- Real World
- Analog front end + ADC
- Processing
- Detector
- Tracker
Sensor outputs vary along the processing chain

<table>
<thead>
<tr>
<th>Virtual Scenario</th>
<th>Signals/Pixels</th>
<th>Processed Signals</th>
<th>Object Detections</th>
<th>Tracks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radar</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lidar</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sensor outputs vary along the processing chain

<table>
<thead>
<tr>
<th>Virtual Scenario</th>
<th>Signals/Pixels</th>
<th>Processed Signals</th>
<th>Object Detections</th>
<th>Tracks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td><img src="image_url" alt="Camera Image" /></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radar</td>
<td><img src="image_url" alt="Radar Image" /></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lidar</td>
<td><img src="image_url" alt="Lidar Image" /></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sensor outputs vary along the processing chain
Sensor outputs vary along the processing chain

- Real World
- Analog front end + ADC
- Processing
- Detector
- Tracker

<table>
<thead>
<tr>
<th>Virtual Scenario</th>
<th>Signals/Pixels</th>
<th>Processed Signals</th>
<th>Object Detections</th>
<th>Tracks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td>![Camera Image]</td>
<td>![Processed Images]</td>
<td>![Object Detection]</td>
<td>![Tracks]</td>
</tr>
<tr>
<td>Radar</td>
<td>![Radar Image]</td>
<td>![Processed Images]</td>
<td>![Object Detection]</td>
<td>![Tracks]</td>
</tr>
<tr>
<td>Lidar</td>
<td>![Lidar Image]</td>
<td>![Processed Images]</td>
<td>![Object Detection]</td>
<td>![Tracks]</td>
</tr>
</tbody>
</table>
Sensor outputs vary along the processing chain.
Many options to bring sensor data to perception algorithms

Perception
- Localization
- Mapping
- Tracking

Planning

Control

Scenario Definition and Sensor Simulation

- Ownship Trajectory Generation
- Actors/Platforms

INS (IMU, GPS) Sensor Simulation

Sensor Data

rosbag data

Multi-object Trackers

Fusion for orientation and position

SLAM

Visualization & Metrics

Perception

Planning

Control
Live data can be augmented for a more robust testbench.
Live data can be augmented for a more robust testbench
Estimate the pose using Monte Carlo Localization

- Perception
  - Localization
  - Mapping
  - Tracking

- Planning

- Control

- Motion Model
  (Odometry readings)

- Sensor Model
  (Lidar scan)

- Particle Filter

- Known Map
What is the world around me?

Egocentric occupancy maps

- Support dynamic environment changes
- Synchronization between global and local maps
What is the world around me?

3D Occupancy Map

Perception
- Localization
- Mapping
- Tracking

Planning

Control

3D map for Autonomous Driving

3D map for UAV motion planning
Where am I in the unknown environment?
Simultaneous Localization and Mapping (SLAM)

Build a map of an unknown environment while simultaneously keeping track of robot's pose.
Simultaneous Localization and Mapping
SLAM Map Builder App (2D only)

App enables more interactive and user-friendly workflow
Simultaneous Localization and Mapping
3D Lidar SLAM

- Perception
  - SLAM
    - Localization
    - Mapping
    - Tracking
- Planning
- Control

Point Cloud Processing → Local Mapping → Loop Closure Detection → Pose Graph Optimization → Map representation

Computer Vision

Navigation

MATLAB EXPO
Autonomous systems can track objects from Lidar point clouds

Track Objects Using Lidar: From Point Cloud to Track List

- Perception
  - Localization
  - Mapping
  - Tracking

Planning

Control

Track surrounding objects during automated lane change
2D radar can be used to track position, size, and orientation.

Perception
- Localization
- Mapping
- Tracking

Planning

Control
Fusing multiple sensor modalities provides a better result
Radar and Lidar fusion can increase tracking performance
Fuse lidar point cloud with radar detections

- Assess missed tracks
- Assess false tracks
- Assess Generalized Optimal Sub Pattern Assignment Metric (GOSPA)
Plan a path from start to destination

- Perception
  - Localization
  - Mapping
  - Tracking

- Planning

- Control

Global Planning

Initial Route to Package

Initial Location

Goal Location

Local Re-planning

Forklift Route to Package

Initial Route to Package

Forklift Route to Package

X [meters]

Y [meters]
Plan a path from start to destination

**Global Planning**
- Path planning algorithms
  - mobileRobotPRM
  - plannerRRT
  - plannerRRTStar
  - plannerHybridAStar

**Behavior Planning**
- High-level decision making
  - Lane keeping/changing
  - Intersection handling
  - Traffic light handling
  - Adaptive cruise control

**Local Re-planning**
- Trajectory generation
  - trajectoryOptimal
  - Frenet

Perception
- Localization
- Mapping
- Tracking

Planning

Control
Urban driving needs planning on multiple levels
Global, behavior, and local planners

Generate optimal trajectories for local re-planning and merge back with the global plan

Perception
- Localization
- Mapping
- Tracking

Planning

Control
Simulate shortest path to change lanes on a highway

- Perception
  - Localization
  - Mapping
  - Tracking

- Planning

- Control

Simulate trajectory generation and the lane change maneuver
Mission planning for UAV leads to last mile delivery

Perception
- Localization
- Mapping
- Tracking

Planning

Control

Flight controller running in external mode on the Raspberry Pi

Waypoint Following
Launch
Orbit

Simulink
QGroundControl

Video Speed: 4x
Choose a path planner based on your application

Sampling-based planners such as RRT*

Use path metrics to compare different paths
Compute control commands for ground vehicles

Compute linear and angular velocity commands for a mobile robot

Perception
- Localization
- Mapping
- Tracking

Planning

Control

Computes angular velocity from this desired curvature
Use Pure Pursuit controller with supervisory logic
Send control commands to the vehicle to follow the planned path

Calculate the steering angle and vehicle velocities to track the trajectories
Control lane change maneuver for highway driving

Longitudinal and Lateral Controllers to adjust the acceleration and steering

Perception
- Localization
- Mapping
- Tracking

Planning

Control
Simulate high-fidelity UAV model with waypoint following

Simulate GPS and IMU sensor models

Waypoint following controller

Approximate High-Fidelity Model with Low-Fidelity Model
Bridging ROS with MATLAB and Simulink: From Algorithm To Deployment

- **ROS bags** (Log files)
- **Data analysis and playback**
- **Standalone node deployment**

**Simulators**
- ROS 2
- ROS Nodes (Software)
- ROS Toolbox

**Desktop prototyping**
- Controls
- Perception
- Planning and decision making

**Hardware**
- **ROS**
- **Desktop prototyping**
- **Standalone node deployment**

**MATLAB EXPO**
Deploy and test sensor fusion and navigation algorithms on hardware.

Perception → Planning → Control

C/C++ Code
HDL Code
GPU Code

Processors
FPGAs
GPUs

MATLAB EXPO
Full Model Based Design Workflow for Autonomous Systems

Verification & Validation

Connect / Deploy

Autonomous Algorithms

Platform

Perceive → Plan & Decide

Sensor Fusion and Tracking Toolbox
Computer Vision Toolbox
Image Acquisition Toolbox
Robotics System Toolbox
Automated Driving Toolbox
Stateflow
Reinforcement Learning Toolbox
Model Predictive Control Toolbox
MATLAB
Simulink
ROS Toolbox
AUTOSAR Blockset
Navigation Toolbox
Code Generation
You can lower risk in your autonomous navigation development

- **Localization**
- **Mapping**
- **Tracking**
- **Perception**
- **Motion Planning**
- **Path Metrics**
- **Path/Waypoint Following**
- **Control**
There are many resources to get started with Tech Talks:

- **Sensor Fusion and Tracking Toolbox**: Design and simulate multisensor tracking and navigation systems.
- **Navigation Toolbox**: Design, analyze, and deploy algorithms for localizing and tracking.
- **Automated Driving Toolbox**: Design, simulate, and test ADAS and autonomous driving systems.

**Contact for Training & Consulting Services**

Exploit the full potential of MathWorks products.
Thank you!
abhishet@mathworks.com