ROS F1/10 Autonomous Racing Simulator

Varundev Suresh Babu and Prof. Madhur Behl
Computer Science | Link Lab | University of Virginia
Varundev Suresh Babu
Ph.D. Candidate
Computer Engineering
University of Virginia
Email: varundev@virginia.edu
What is F1/10 [f1tenth]?
Build. Drive. Race. 

Perception. Planning. Control
BUILDD
Construct your vehicle using our starter kit.

CODE
Learn to drive your vehicle autonomously.

RACE
Register to Compete
Similar dynamics, different parameters

TRAXXAS XO-1 vs Tesla Model S
Similar sensors, different scale
Sensor Integration

- LiDAR
- Camera
- IMU
- IR Depth Cameras
Sensor Integration

LiDAR  Camera  IMU  IR Depth Cameras

Wi-fi Telemetry  Onboard Computer  Motor Controller
FOLLOW THE GAP METHOD

Follow the gap method
Simple obstacle avoidance

MODEL PREDICTIVE CONTROL

Generated paths
Max. cost (obstacle)
Drivable region
Drivable region
Dynamic obstacle
Medium cost
Unexplored space

END-TO-END DNN

FPV data annotation
Predicted steering (blue)
Ground Truth (green)

V2V COLLABORATION

F1/10 FPV
F1/10 RGB FPV

Collaborative behavior automaton

I'm at a stop sign

Env. Vehicle Path
Environment vehicle
Ego vehicle

Follow trajectory
Guard 1: Environment vehicle not in clear zone
Guard 2: Environment vehicle in clear zone
Guard 3: Environment vehicle enters clear zone
Guard 4: Ego vehicle enters clear zone
Exit roundabout
Merge
LANE KEEPING ASSIST

Lane detection pipeline

Original image
Hough transform
Threshold masking

COMPUTER VISION EXAMPLE

Optical flow

FlowNet outputs

F1/10 online semantic segmentation

LOCALIZATION AND MAPPING

Currently tested On F1/10

Hector SLAM
AMCL
Google Cartographer
ORB-SLAM

Sensor Scan region
Track boundaries
Localized vehicle
Travel path

Global planning using rviz

Vehicle chassis
Laser Scanner

ROS transform frame
1st F1/10 Race: Oct 2016, Pittsburg, ES-Week

2nd F1/10 Race: Apr 2018, Porto, CPS-Week

3rd F1/10 Race: Oct 2018, Torino, ES-Week

4th F1/10 Race: Apr 2019, Montreal, CPS-Week
Head-to-head Autonomous Racing
4th F1/10 International Autonomous Racing Competition
Limitations

Hardware cost - $3000

Slow Algorithm Development
From design to hardware implementation
Simulator Requirements

- Ubuntu 18.04 (bionic)
- ROS Melodic
- Gazebo
Instructions on f1tenth.dev

roslaunch f1tenth-sim simulator.launch
Simulator Elements

Racecar

Race Track

Algorithms
Simulator Architecture

Front End
- Route Planning Control Plugin
- Vehicle Model Sensor Plugin

ROS
- Navigation Stack
  - TEB planner
  - Pure Pursuit
- Localization
  - AMCL
  - GPU Particle Filter
- Controller Manager
- TF Broadcaster
  - Odom → Base
  - Base → Laser
  - Base → Camera
- State Estimation
  - Precise Odom

Simulator Back End
- Simulated Environment
  - Race Track
  - Flag
  - Post & Start Line
  - Lap Counter
- Simulated Racecar
- Data Visualization

Simulator Architecture
Navigation and Control Algorithms

ROS

Navigation Stack
- TEB planner
- Pure Pursuit

Localization
- AMCL
- GPU Particle Filter

Controller Manager

TF Broadcaster
- Odom → Base
- Base → Laser
- Base → Camera

State Estimation
- Precise Odom

Front End
- Route Planning Control Plugin
- Vehicle Model Sensor Plugin

Simulator Back End
- Simulated Environment
  - Race Track
  - Flag Post & Start Line
  - Lap Counter
- Simulated Racecar

Data Visualization

Navigation and Control Algorithms
Gazebo and Visualization

Front End
- Route Planning Control Plugin
- Vehicle Model Sensor Plugin

ROS
- Navigation Stack
  - TEB planner
  - Pure Pursuit
- Localization
  - AMCL
  - GPU Particle Filter
- Controller Manager
- TF Broadcaster
  - Odom → Base
  - Base → Laser
  - Base → Camera
- State Estimation
  - Precise Odom

Simulator Back End
- Simulated Environment
  - Race Track
  - Flag Post & Start Line
  - Lap Counter
- Simulated Racecar

Data Visualization
F1/10 Autonomous Racecar Platform

Powertrain (with Ackermann dynamics)

4 Wheel Independent Drive (with hard differential constraints enforced by control plugin)

Size & Performance Replicated
CAD, xacro, urdf

Simulated F1/10 Racecar
- Customizable color
- Multiple sensors

Modular Construction
- LiDAR on Top
- Camera on the hood

Gazebo Sensor Plugins

Racecar Model
Primary Navigation Sensor
2D Scanning LiDAR

- Hokuyo sensor plugin
- LaserScan.msg message type
- Range (4.0m to 30.0m)
- Scan Angle (180 to 360 degrees)
- Position changed in URDF
Modular Design

Secondary Navigation Sensors

- Stereo Camera
- Depth Camera
- IMU
- Front/Rear Two Point Radar
Modular Design

Racecar Chassis
(Shell, powertrain,
Collision sensor, controller)

- Collision sensor
- Rear Wheel/All wheel drive
- Speed set as RPM or m/s
- Differential enable/disable
- Energy meter and bank with boost
Racecar Control Nodes

tf Nodes (static & dynamic)

/car_name/base_laser_link
/car_name/base_odom_link

Command, Control & Feedback Node

/car_name/control_plugin

Node Graph
Control Plugin

Command & Control

Message Type:
AckermannDrive.msg
Command & Control

Steering Column Controller

Ackermann Steering Geometry

Float64.msg

Message Type: AckermannDrive.msg
Control Plugin

Command & Control

Message Type: AckermannDrive.msg

Wheel RPM Controller

PID Speed Control

Float64.msg
tf-Tree

- **map**
  - Broadcaster: `/map_world_link`
  - Average rate: 10.191 Hz
  - Most recent transform: 80.917 (0.008 sec old)
  - Buffer length: 4.808 sec

- **odom**
  - Broadcaster: `/car_1/control_plugin`
  - Average rate: 50.412 Hz
  - Most recent transform: 80.873 (0.036 sec old)
  - Buffer length: 4.860 sec

- **car_1_base_link**
  - Broadcaster: `/car_1/base_laser_link`
  - Average rate: 49.416 Hz
  - Most recent transform: 80.915 (-0.006 sec old)
  - Buffer length: 4.877 sec

- **car_1_laser**
Racecar Object Creation

<!-- spawn car using the set global parameters -->

<include file = '${(find f1tenth-sim)/launch/simulator.launch}'>
  <arg name = 'car_name' value = '${(arg car_name)}'/>
  <arg name = 'paint' value = '${(arg car_paint)}'/>
  <arg name = 'run_gazebo' value = '${(arg run_gazebo)}'/>
  <arg name = 'x_pos' value = '${(arg x_pos)}'/>
  <arg name = 'y_pos' value = '${(arg y_pos)}'/>
</include>
<!-- spawn car using the set global parameters -->

<include file = '${(find f1tenth-sim)/launch/simulator.launch'}>

<arg name = 'car_name' value = '${(arg car_name)'/>  

<arg name = 'paint' value = '${(arg car_paint)'/>  

<arg name = 'run_gazebo' value = '${(arg run_gazebo)'/>  

<arg name = 'x_pos' value = '${(arg x_pos)'/>  

<arg name = 'y_pos' value = '${(arg y_pos)'/> </include>
Racecar Object Creation

<!-- spawn car using the set global parameters -->

<include file = '${(find f1tenth-sim)/launch/simulator.launch}'>

<arg name = 'car_name' value = '${(arg car_name)}'/>

<arg name = 'paint' value = '${(arg car_paint)}'/>

<arg name = 'run_gazebo' value = '${(arg run_gazebo)}'/>

<arg name = 'x_pos' value = '${(arg x_pos)}'/>

<arg name = 'y_pos' value = '${(arg y_pos)}'/></include>
Racecar Object Creation

<!-- spawn car using the set global parameters -->

<include file = '${(find f1tenth-sim)/launch/simulator.launch'}>

<arg name = 'car_name'
value = '${(arg car_name)'} />
<arg name = 'paint'
value = '${(arg car_paint)'} />
<arg name = 'run_gazebo'
value = '${(arg run_gazebo)'} />
<arg name = 'x_pos'
value = '${(arg x_pos)'} />
<arg name = 'y_pos'
value = '${(arg y_pos)'} />
</include>

Circuit-Breaker (keeps only one session active)
<!-- spawn car using the set global parameters -->

<include file = '${(find f1tenth-sim)/launch/simulator.launch}'>
  <arg name = 'car_name' value = '${(arg car_name)}'/>
  <arg name = 'paint' value = '${(arg car_paint)}'/>
  <arg name = 'run_gazebo' value = '${(arg run_gazebo)}'/>
  <arg name = 'x_pos' value = '${(arg x_pos)}'/>  
  <arg name = 'y_pos' value = '${(arg y_pos)}'/>  
</include>
Namespace resolution handled by ROS
Capable of implementing different controllers
Multiple Racecars (tf-frames)

- **map**
  - Broadcaster: /map_world_link
  - Average rate: 10.196 Hz
  - Most recent transform: 51.037 (-0.096 sec old)
  - Buffer length: 4.904 sec

- **odom**
  - Broadcaster: /car_2/control_plugin
    - Average rate: 50.274 Hz
    - Most recent transform: 50.913 (0.028 sec old)
    - Buffer length: 4.933 sec
  - Broadcaster: /car_1/control_plugin
    - Average rate: 50.546 Hz
    - Most recent transform: 50.926 (0.015 sec old)
    - Buffer length: 4.946 sec

- **car_2_base_link**
  - Broadcaster: /car_2/base_laser_link
    - Average rate: 49.909 Hz
    - Most recent transform: 50.945 (-0.004 sec old)
    - Buffer length: 4.949 sec

- **car_1_base_link**
  - Broadcaster: /car_1/base_laser_link
    - Average rate: 49.960 Hz
    - Most recent transform: 50.960 (-0.019 sec old)
    - Buffer length: 4.964 sec

- **car_2_laser**
- **car_1_laser**
Race Track creation process

Sketch a 2D track

Convert to 3D mesh

Export as STL/DAE
Race-Track Setup (Gazebo View)

- Race Car
- Drivable Area
- Track Bounds
Race-Track Setup (rviz View)
Mapping – Hector SLAM

roslaunch f1tenth-sim simulator.launch
roslaunch f1tenth-sim mapping.launch
Localization – GPU Particle Filter

roslaunch f1tenth-sim simulator.launch
roslaunch f1tenth-sim localization.launch
Localization
(Implemented for multiple vehicles)

Odometry sources
- Gazebo (available, default option)
- Laser Scanmatcher (available, implemented)
- Virtual Wheel Encoders (work in progress)

Provided by
- AMCL (default, implemented)
- GPU Particle Filter (available, tested)
Motion Planning – ROS Navigation with TEB Planner

roslaunch fitenth-sim navigation.launch
Multiple Independent Navigation
Racing Algorithms (Single Racecar)

roslaunch f1tenth-sim purepursuit_one_car.launch
Racing Algorithms (Multiple Racecars)
Get Started

f1tenth.dev

varundev@virginia.edu
**F1/10** Autonomous Racing Simulator is now open-source and available!

**Step 1)**
```
docker run -it --rm -p 6080:80 madhurbehl/f1tenth
```

**Step 2)**
Visit 127.0.0.1:6080 in your favorite browser

**Algorithms Supported:**
- Follow-the-gap planner
- Hector-SLAM mapping
- Adaptive Monte Carlo Localization
- Time-Elastic Band Local Planner

Visit madhurbehl/f1tenth on DockerHub or click on the icons below