AA 274A
Principles of Robot Autonomy I

Open-source Automated Driving Stack „Autoware“
Today’s lecture

• Aim
  • Overview Autoware (Autoware.AI, Autoware.Auto), software architecture
  • Simulation environments for Autoware
  • Integration of Autoware into a research vehicle
  • Hands-on
    • Installation, development environment
    • Demos: Localization, object detection, path planning

• Readings
  • https://www.autoware.org/
  • https://gitlab.com/autowarefoundation/autoware.ai/autoware/wikis/Installation
  • https://gitlab.com/autowarefoundation/autoware.ai/autoware/wikis/home
Overview Autoware

• Other Automated driving stacks
• Autoware.AI (ROS1), Autoware.Auto (ROS2)
• Software architecture
  • General overview
  • Localization
  • Object detection
  • Path planning
Autonomous software stack solutions

Most relevant software stacks in 2019

- DriveWorks (Nvidia)*
- Apollo*
- **Autoware**
- EB robinos & EB robinos Predictor (Elektrobit)
- OpenPilot (comma.ai)

* will be briefly introduced
Nvidia DriveWorks

https://developer.nvidia.com/drive/drive-software
Nvidia Drive Constellation Architecture
Apollo

Android of the autonomous driving industry, but more open and powerful.
Apollo / Software modules

- Data Pipeline
- Perception
- Planning
- Control
- Prediction
- Map Engine
- Simulation
Apollo in action

Unprotected Left Turn
Autoware

• Autoware was started 2015 by Shinpei Kato at Nagoya University.
• "All-in-One" open-source software for autonomous driving technology.

• Autoware Foundation launched in 2018.
• Non-profit organization supporting open-source projects enabling self-driving mobility.
• It is based on ROS 1 and available under Apache 2.0 license

• It contains the following modules:
  • **Localization** is achieved by 3D maps and SLAM algorithms in combination with GNSS and IMU sensors.
  • **Detection** uses cameras and LiDARs with sensor fusion algorithms and deep neural networks.
  • **Prediction** and **Planning** are based on probabilistic robotics and rule-based systems, partly using deep neural networks as well.

• The output of Autoware to the vehicle is a twist of velocity and angular velocity (yaw rate).
Software modules

Sensing
- LIDAR
- GNSS
- Camera
- IMU
- Radar
- Etc.

Localisation
- Absolute (i.e. lat, long, etc)
- Relative to other agents

External
- Map

Perception / Situation Awareness
- Computer Vision
- ML / DL
- KFs, PFs

Obstacle detection, object classification (traffic lights, signals), lane detection

Navigation
- Global planner (route A-B)
- Local planner (obstacle free path)
- Decision making

Actuation
- Path following
- PIDs, etc: vehicle dependent

Simulation

V2X
- CAM
- DENM
- Future specs
Abstraction layers
• An interface project for Autoware to be extended with proprietary software and third-party libraries in a reliable manner.

• Include device drivers for sensors, by-wire controllers for vehicles, and hardware-dependent programs for SoC boards.

• Provides a hardware reference platform with tools, unified interface design and test framework.

• Enables the integration of member company's solutions onto platforms which support the Autoware.Auto and Autoware.AI software stack.
• Re-implementation of Autoware.AI
• ROS2 based
• Clearly defined APIs and interfaces for the different modules
• State of the Art development process CI/CD
  • Pull request reviews, pull request builds
  • Comprehensive documentation
  • 100% code coverage
  • Coding style guide
  • Managed by an open source community manager
• Will initially address Autonomous Valet Parking and Autonomous Depot Maneuvering as example use cases.

• Autoware.Auto will allow mapping of a parking lot, creation of a map for autonomous driving and autonomous driving on the parking lot.
Autoware outlook - commercial use – APEX.AI

1. User Application
   - First release in 2010
   - SDK for robotic systems
   - Designed for research
   - Open Source: BSD

2. ROS 1
   - Autonomous
   - Autoware.AI

3. ROS 2
   - Optimized
   - Autoware.Auto

4. Apex.OS
   - Certified
   - Apex.Autonomy

5. Improvements ROS 1 > ROS 2
   - Improved code quality
   - Smaller and more optimized code
   - Standardized middleware
   - Security on data flow level
   - Testing and documentation

6. Improvements ROS 2 > Apex.OS
   - Production code quality
   - Hard real-time support
   - Real-time logger
   - System safety
   - System security
   - Complete documentation
   - Support for automotive ECUs and automotive sensors
   - Functional safety certification (ISO 26262, SEooC, up to ASIL-D)
   - Generation of binaries, secure signing, shipping
   - 24/7 customer support
Simulation environments for Autoware

- Carla
- LGSVL
- Gazebo
- Autoware simulator
Carla

- Camera, lidar, GPS, ground truth
- Autoware, Apollo interface
- Unreal Engine
- Open Drive
- Scenario modelling
- Detailed camera model
LG SVL Simulator

- Camera, lidar, GPS, ground truth
- Autoware, Apollo interface
- Unity engine
- Road editor
- Radar simulation
- GPU optimized lidar model
Gazebo

- Willow garage project, since 2012 Open Source Robotics Foundation
- ROS Interface
- Different sensor models
  - Camera
  - Lidar
- Vehicle model

+ not automotive specific
+ large community
- custom engine
- modelling other traffic participants
Autoware Simulator

• Focus: Path Planning algorithms
• Function development only based on ground truth data
• Simulation of 5 other vehicles

• Simple vehicle models
Integration of Autoware into a research vehicle
Integration of Autoware into a research vehicle

• Vehicle hardware components

• Mapping of road network

• Required software components from Autoware
Hardware components

**x86 Computer**
Ubuntu 16.04 / ROS Kinetic

**Localization:**
Novatel RTK-GPS / ProPak6
Positioning < 5cm, 100Hz via TCP/IP

1x Long Range Radar
Continental ARS408

1x Lidar
Ouster OS1-64

1x Mobileye

2x FLIR Cameras

11/13/2019
Hardware architecture

- **Novatel ProPak6 RTK-GPS**
  - Ethernet

- **Ouster OS1 / 64**
  - Ethernet

- **ADAS-KIT (Dataspeed)**
  - CAN

- **Mobileye**
  - CAN

- **Radar ARS408**
  - CAN

- **RGB Camera**

- **USB-CAN Gateway (Dataspeed)**
  - 4xCAN
  - USB

- **Gigabit Router**
  - Ethernet

- **x86 Computer**
  - ROS
  - USB

- **Cockpit Monitor**
  - USB, HDMI

- **WIFI/LTE Router**
Software components

- **Map**
  - HD map creation
  - Global planning

- **Localization**
  - Lidar localization
  - Local Planning

- **Object Detection**

- **Path Planning**
Mapping of Road Network

• Mapping for localization (without RTK-GPS)
  • Lidar with 64 layers (Ouster OS1-64)
  • NDT mapping
  • [link](https://tools.tier4.jp/feature/vector_map_builder/)

• HD map for path planning
  • Browser based tool for mapping (Tier IV)
  • Current data format: Asian Vector Map
  • Future data format: [Lanelet2](#), [OpenDrive](#)
Localization

• Normal distributions transform (NDT) matching
• Lidar based / 64 Layers / 20 Hz / Voxel Grid 1m

https://gitlab.com/autowarefoundation/autoware.ai/core_perception/tree/master/lidar_localizer/nodes/ndt_matching

11/13/2019
Object Detection

Raw Sensor Data

Ground / No Ground

Point Cloud Clustering

Object Tracking
Object Detection / Classification

Full HD RGB Camera / Yolo3: Neural Network for Object Detection
Path Planning / Trajectory Tracking

Global Planner

Local Planner

Local Planner

Trajectory Tracking
Vehicle Interface

• ADAS-Kit Dataspeed Inc.
• Universal Lat/Lon Controller Interface

➢ Controller for drive-by-wire interface (execution on Dataspeed ECUs)
➢ Velocity / yaw or curvature control (/twist_cmd)
Hands-on

- Installation on PC
- Development environment
- Autoware Demos
  - Localization (with recorded data from TU/Stanford campus)
  - Object detection (with recorded data)
  - Path planning (with Autoware simulator)
Installation

• Recommended System Specifications for complete stack
  • Number of CPU cores: 8, Nvidia GPU
  • RAM size: 32GB, Storage size: 64GB+

*Depends extremely which components are used from the stack. Runs also in a virtual machine.*

• Source Build

• Docker (recommended)

https://gitlab.com/autowarefoundation/autoware.ai/autoware/wikis/Installation
Path planning

- **Global planner** [op_global_planner]

- **Local planner** [op_trajectory_generator, op_motion_predictor, op_trajectory_evaluator, op_behavior_selector]
  - Input: /tracked_objects, /global_path
  - Output: /final_waypoints

- **Trajectory Tracking** [pure_pursuit or mpc_follower, twist_filter]
  - Input: /final_waypoints
  - Output: /twist_cmd

- **Autoware Simulator** [wf_simulator]
  - Input: /twist_cmd
  - Output: /simulated_objects
Localization pipeline

- **Map loader** [points_map_loader]
  - PCD loader from map

- **Voxel Grid Filter** [voxel_grid_filter]
  - Downsampling lidar data
  - Leaf size: 2m (60MB/s → ~1MB/s)

- **Lidar based localization** [ndt_matching]
  - NDT matching
  - Input: /filtered_points, /vehicle/twist
  - Output: /ndt_pose

- **EKF Localization Fusion** [ekf_localizer]
  - Input: /ndt_pose, /vehicle/twist
  - Output: /ekf_pose_with_covariance
Object detection

• Ray ground filter [ray_ground_filter]
  • Separation ground / no ground

• Point cloud clustering [lidar_euclidean_cluster_detect]
  • NDT matching
  • Input: /points_no_ground
  • Output: /points_cluster

• Cluster tracker [lidar_kf_contour_track]
  • Input: /points_cluster
  • Output: /tracked_objects
Thanks for your attention!
Questions?

Daniel Watzenig, Markus Schratter
daniel.watzenig@v2c2.at
markus.schratter@v2c2.at