McMaster Autonomous Electric Vehicle (MacAEV)

ELECENG 3EY4: Electrical Systems Integration Project

Shahin Sirouspour

Winter 2024

Plan for Second Part of Course

- Explore autonomous driving concepts with McMaster AEV
- Lectures, tutorials and laboratories are organized around weekly milestones
- Lectures and tutorials introduce theoretical concepts at high level with some technical details left out due to time constraints
- Resources and references are made available so students can explore and develop deeper understanding of these concepts on their own
- Lectures and tutorial time is also used to go over project expectations for each week
- Lab hours are used to help students with project implementation issues, and to assess their progress

Key Points to Consider

- This is a project-based course, so you should expect to spend time outside regular class and lab hours towards achieving the objectives
- There are many steps in development and implementation that can be carried out at home, and you are strongly encouraged to do so
- In fact, it is quite likely that you would not be able to finish everything if you were to work exclusively during the lab time
- We will require a group report for each lab. Due two weeks from the last day of the lab.

Milestones

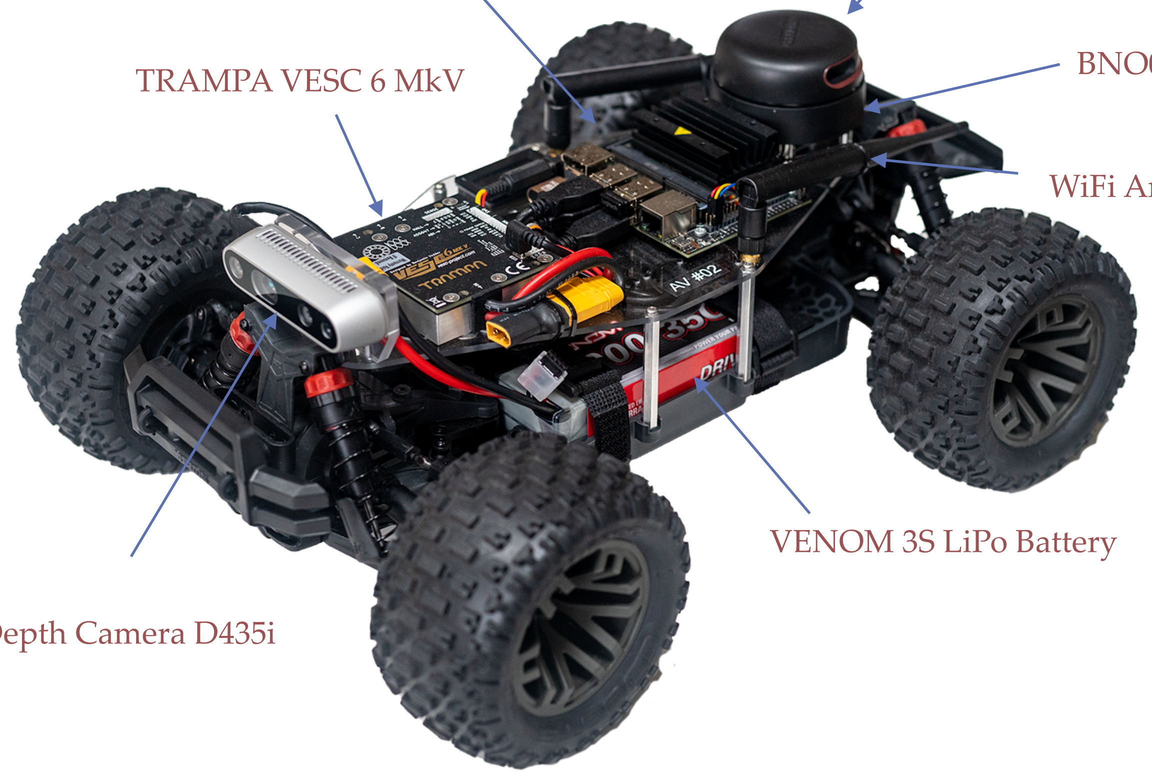
- **Lab 6 (Feb. 26-March 1):** Setting Up Simulation Environment and Manual Driving of McMaster AEV
- **Lab 7 (March 4-15)** Localization and Mapping with MacAEV
- **Lab 8 (March 18-22):** Autonomous Driving Using Reactive Wall-Following Method
- **Lab 9 (March 25-29):** Autonomous Driving Using Virtual Separating Barriers
- **Lab 10 (April 1-5):** Improving Autonomous Driving using RGB-D Camera Data (Bonus Activity)

Bonus Project Activity

- All structured laboratory activities use the on-board **LiDAR** as the primary source of information for vehicle assisted and self-driving functionalities
- LiDAR can only see things in its plane of scan
- You can earn a bonus grade of **6%** of your total grade by integrating the data from the on-board **RGB-Depth** camera into vehicle operation
- Modify self-driving algorithm from Lab 9 to integrate the point clouds obtained from RGB-Depth camera

McMaster AEV

Vehicle built on a 1/10th scale RC hobby platform (ARMA GRANITE 4x4 link)



Onboard Computer

NVIDIA Jetson Nano

GPU	128-core Maxwell
CPU	Quad-core ARM A57 @ 1.43 GHz
Memory	4 GB 64-bit LPDDR4 25.6 GB/s
Storage	microSD (not included)
Video Encode	4K @ 30 | 4x 1080p @ 30 | 9x 720p @ 30 (H.264/H.265)
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 18x 720p @ 30 (H.264/H.265)
Camera	2x MIPI CSI-2 DPHY lanes
Connectivity	Gigabit Ethernet, M.2 Key E
Display	HDMI and display port
USB	4x USB 3.0, USB 2.0 Micro-B
Others	GPIO, I^2C, I^2S, SPI, UART
Mechanical	69 mm x 45 mm, 260-pin edge connector

NVIDIA Jetson Nano

- Nvidia JetPack SDK link
- Jetson Linux Driver Package (L4T)
- OS: Linux kernel 4.9, filesystem based on Ubuntu 18.04
- **TensorRT:** Built on Nvidia CUDA, is a high-performance deep learning inference runtime for image classification, segmentation, and object detection neural networks.
- **cuDNN:** CUDA Deep Neural Network library provides provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers
- **CUDA:** Toolkit provides a development environment for GPU-accelerated applications
- **Computer Vison:** Computer Vision / Image Processing algorithms implemented on PVA (Programmable Vision Accelerator), GPU and CPU. Supports **OpenCV** (Open-source library for computer vision) and **VisionWorks** (a software development package for Computer Vision (CV) and image processing)
- **Multimedia API:** Camera Application API (low-level frame-synchronous API for camera applications, with per frame camera parameter control, multiple camera support, and EGL stream outputs). Sensor driver API (enables video decode, encode, format conversion and scaling functionality)

The Robot Operating System (ROS)

- "The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source." link
- You have already installed ROS 1 long-term distribution Melodic which primarily targets the Ubuntu 18.04 (Bionic) release
- ROS provides a communication and modular application development framework targeting robotics applications
- All application developments in the remainder of the course will be based on ROS using Python and some C++ programing languages, as needed

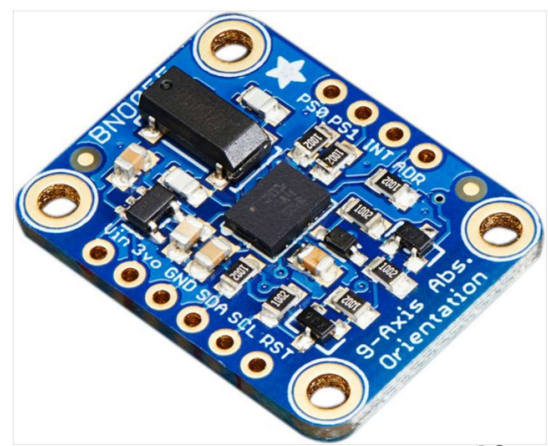
Sensing Modules

RPLIDAR A2 LIDAR

- SLAMTEC RPLIDAR A2 LiDAR 2D 360° [link]
- Pulse-modulated infrared laser signal
- Uses laser triangulation ranging principle
- Works indoor and outdoor (without direct sun exposure)
- Mechanical Frequency: 5-15Hz
- Measuring Range: 0.2m-16m
- Sampling Frequency: 8k/sec
- Angular Resolution: 0.9°
- Range Resolution
  - $\leq 1\%\ of\ the\ range\ (\leq 12m),\ \leq 2\%\ of\ the\ range$ (12m~16m)
- Accuracy
  - $\leq 1\%\ of\ the\ range\ (\leq 3m)$
  - $2\%\ of\ the\ range\ (3-5m)$
  - $2.5\%\ of\ the\ range\ (5-16m)$

Adafruit BNO055 IMU

- 9 DOF Inertial Measurement Unit (IMU) [link]
- Combines a MEMS accelerometer, magnetometer, and gyroscope on a single die
- ARM Cortex-M0 based processor fuses and filters sensory information to produce:
  - Absolute Orientation (Euler angles or Quaternion at 100 Hz)
  - Angular Velocity Vector 3x1 (100 Hz)
  - Acceleration Vector 3x1 - gravity + linear acceleration (100 Hz)
  - Linear Acceleration Vector 3x1 (100 Hz)
  - Magnetic Field Strength Vector 3x1 (20 Hz)
  - Gravity Vector 3x1 (100 Hz)
  - Ambient Temperature (1 Hz)



Intel D435i Depth Camera

- Combines an RGB camera, an active infrared stereoscopic depth camera and IMU link
- Built-in Intel RealSense Vision Processor D4

Use environment:
Indoor/Outdoor

Ideal range:
.3 m to 3 m

Image sensor technology:
Global Shutter

RGB frame resolution:
1920 × 1080

RGB sensor FOV (H × V):
69° × 42°

RGB frame rate:
30 fps

RGB sensor resolution:
2 MP

RGB sensor technology:
Rolling Shutter

Depth technology:
Stereoscopic

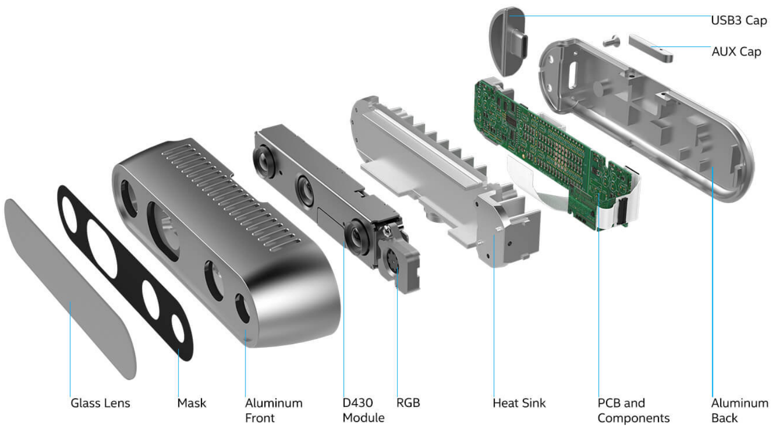
Depth Field of View (FOV):
87° × 58°

Minimum depth distance (Min-Z) at max resolution:
~28 cm

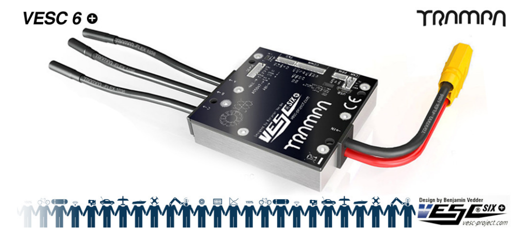
Depth output resolution:
Up to 1280 × 720

Depth frame rate:
Up to 90 fps

Depth Accuracy:
<2% at 2 m[1]



TRAMPA VESC 6 Mark V



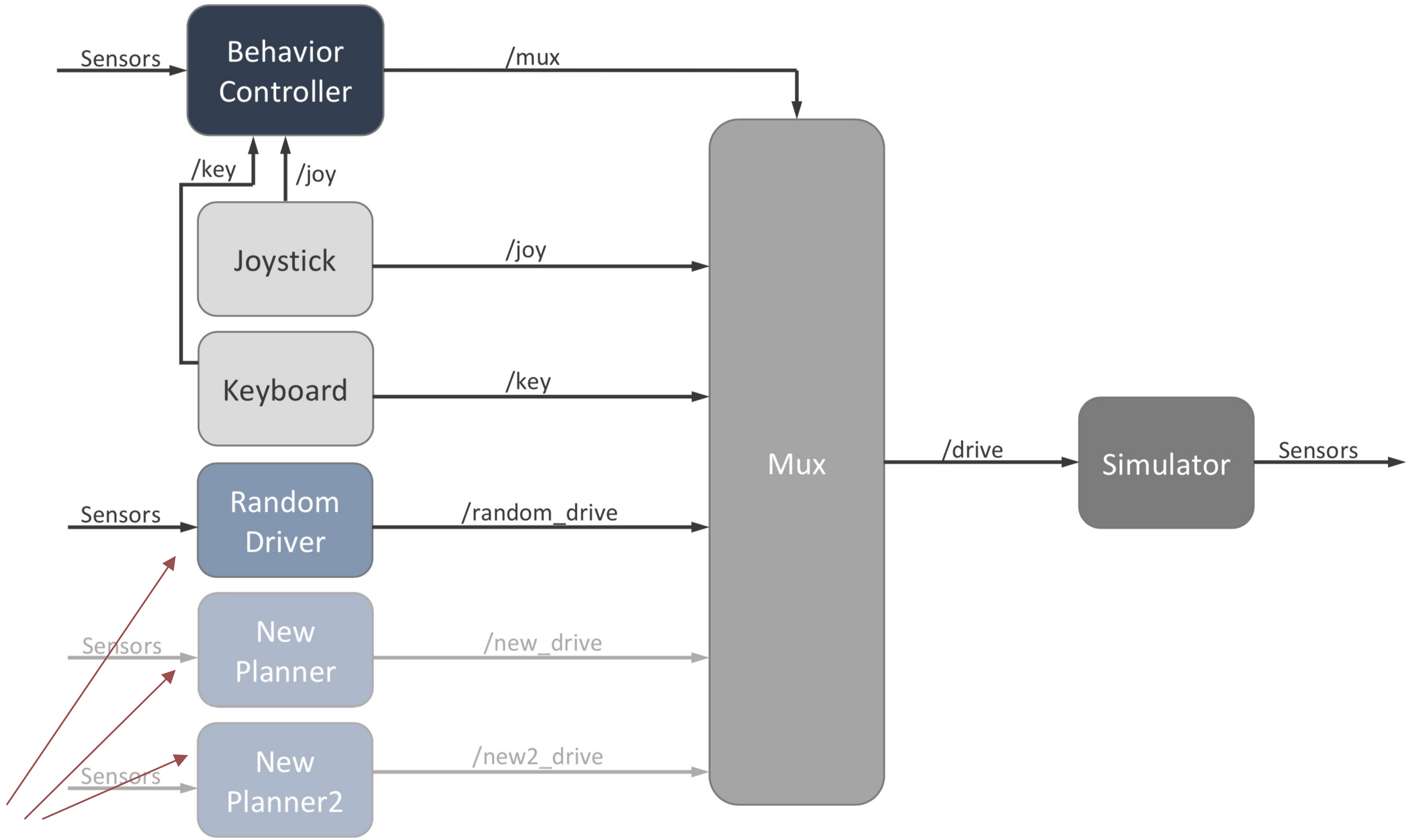
- Can operate in BLDC and FOC modes. We use it in FOC mode due to smoother and more efficient operation
- Built-in safety protection for overcurrent and LiPo battery low voltage
- Initial set-up and tuning up of the parameters with VESC tool
- Provides an estimation of motor velocity for use in wheel odometry
- Connects to Jetson Nano via USB port
- Powered by 3S LiPo battery
- Provides PPM control signal for steering servomotor
- ROS based drivers are used to communicate with VESC
- Can incorporate motor position sensors (not used here)

Software Architecture

F1/10 Simulator

- A lightweight 2D simulator of UPenn F1TENTH Racecar. The simulator can be built with ROS or as a standalone C++ application
- We will build on F1/10 Simulator package available on GitHub link
- We will make changes as necessary to accomplish project-specific objectives
- Using this code structure makes the transition from simulation to experiment seamless
- Control algorithms and codes will be developed and tested on the simulation platform first
- This allows for safe debugging and performance evaluation before testing new algorithms on the vehicle
- Same control code can then be used for driving the actual vehicle

F1/10 Simulator



Source: https://github.com/f1tenth/f1tenth_simulator

Modified Architecture