Vision and Image Sciences Laboratory

SLAM for Formula Driverless

Lital Guy
Maxim Aslyansky

Supervisor:
Eli Appelboim

Winter 2018-2019

Project overview

Formula Student is a student engineering competition where student teams from around the world design, build, test, and race a small-scale formula style racing car.

Our contribution to the group:

Reconstruct the race track and estimate the vehicle position on the map while driving (SLAM)

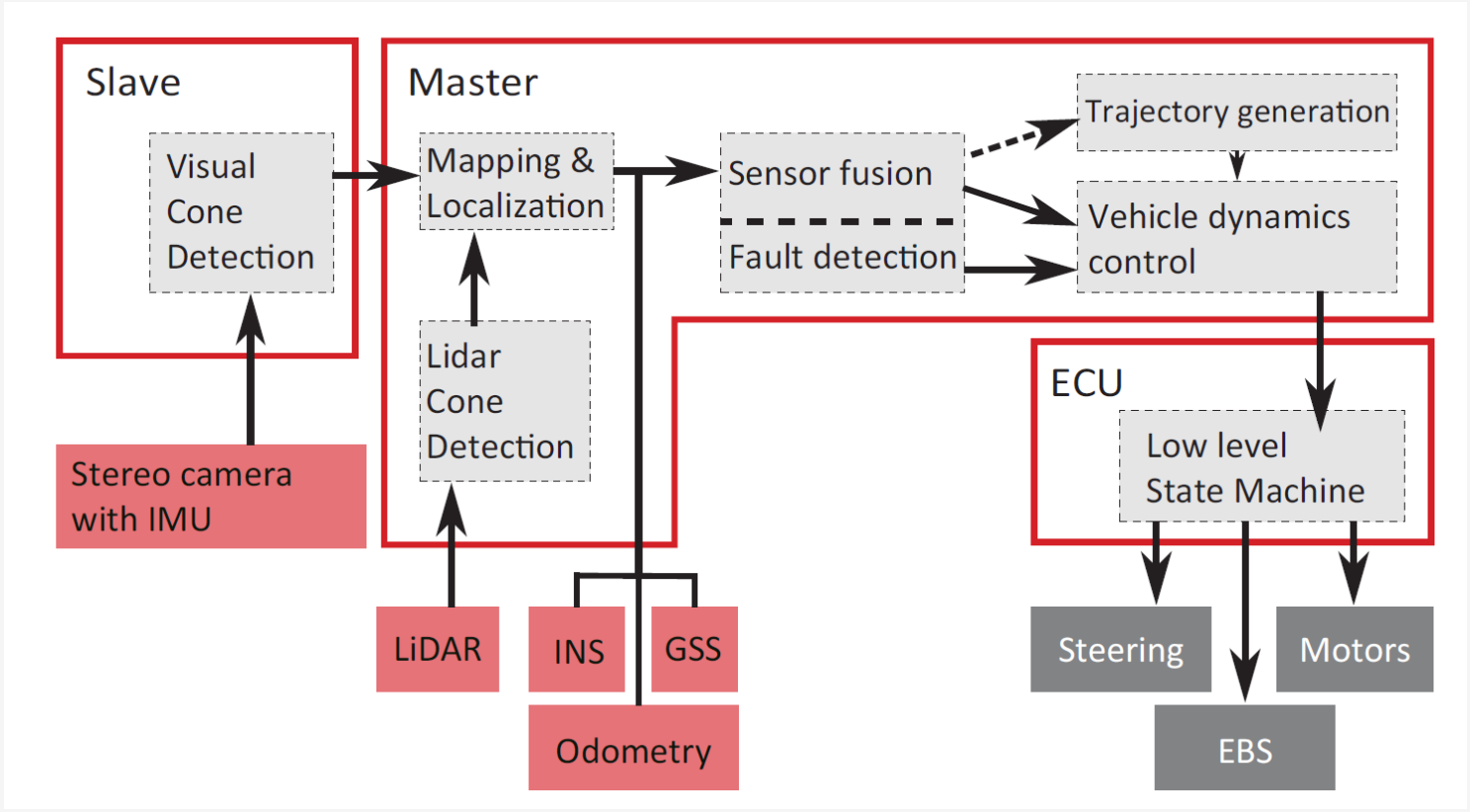
Vision and Image Sciences Laboratory

Project Goal

- Mapping is crucial to achieve high speed
  - During the first loop environment map is built
  - Allows to significantly improve speed by planning beforehand
- Main challenges
  - The algorithm must run in real time (low latency and concurrency issues)
  - SLAM should be robust to fast motions and tracking failures
  - Sunlight, uneven road conditions, missing cones

AMZ driverless in-depth overview

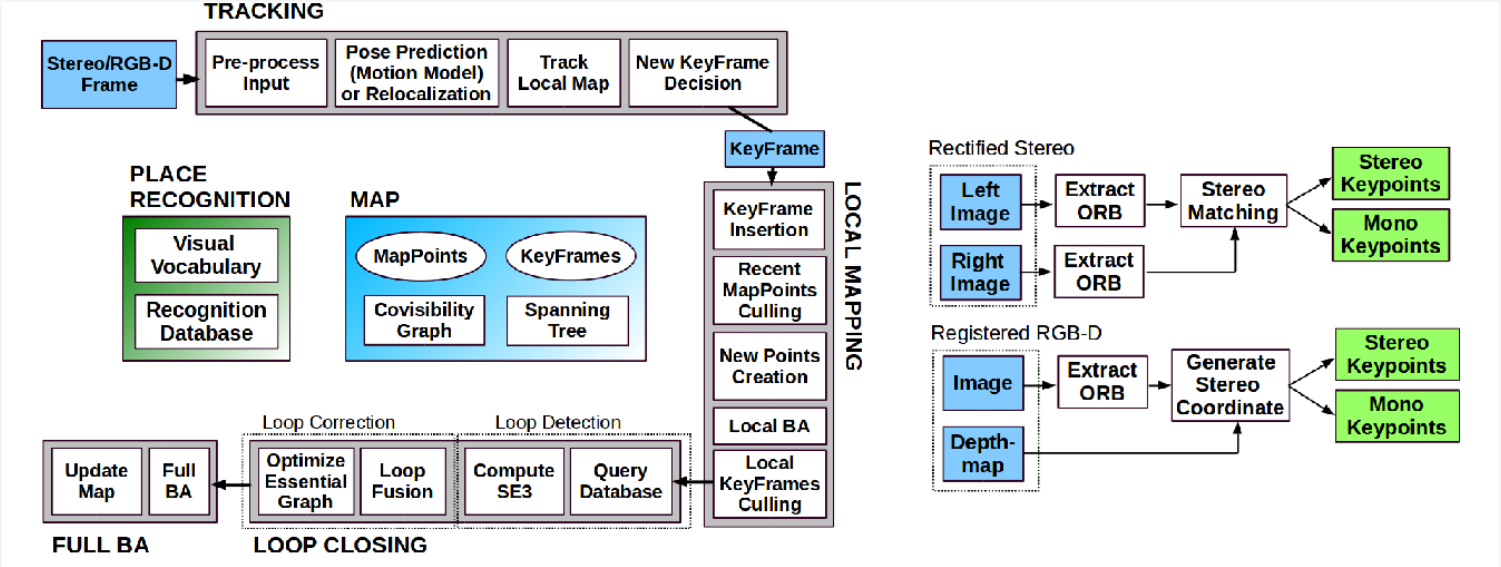
- System Diagram
- Two SLAM pipelines:
  - visual SLAM (+cones)
  - LIDAR SLAM (cones only)
- Fused by Kalman Filter
- Efficient vehicle control



Vision and Image Sciences Laboratory

Visual SLAM

- SERVO is robust extension to ORB-SLAM 2 used by AMZ Driverless team
- First ROVIO (robust visual-inertial odometry) is used to estimate the position
- ORB-SLAM 2 is initialized with estimated pose and IMU measurements
- ROVIO landmarks and cones integrated into ORB-SLAM as well
- ORB-SLAM modified for better robustness to fast motions

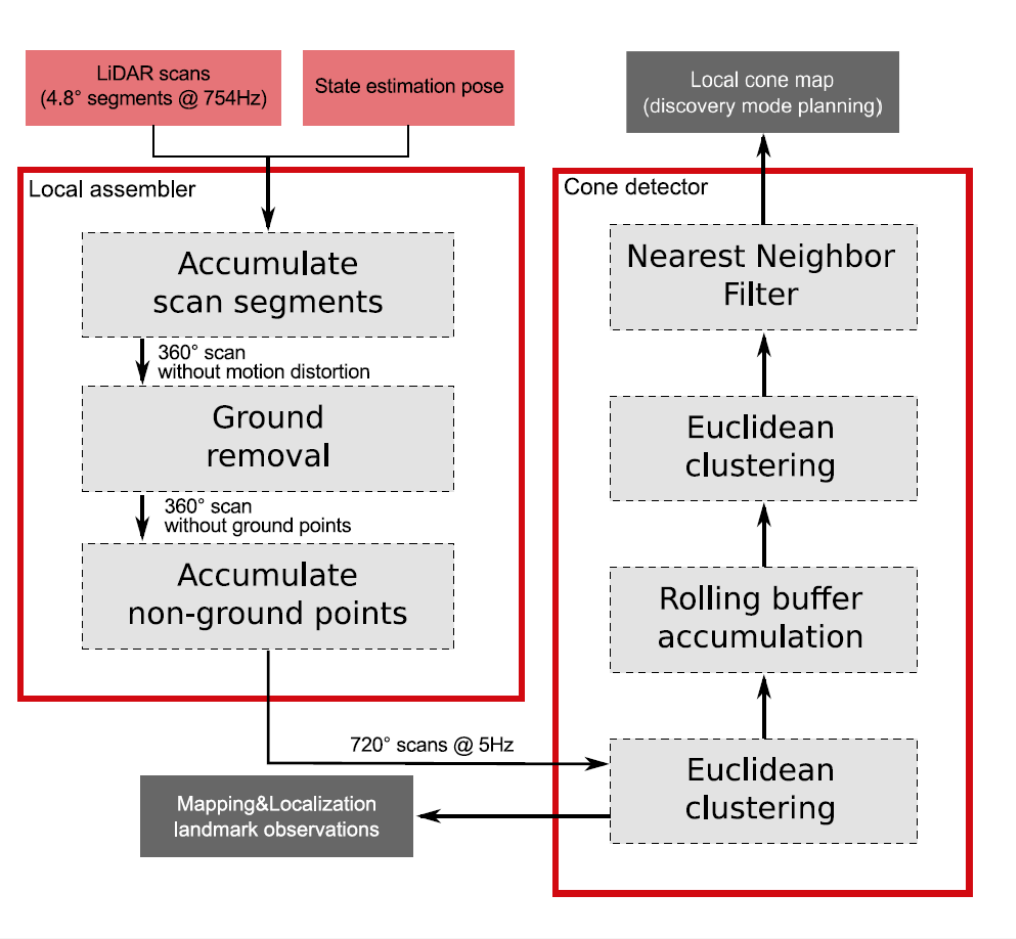


VISL Vision and Image Sciences Laboratory

LIDAR SLAM

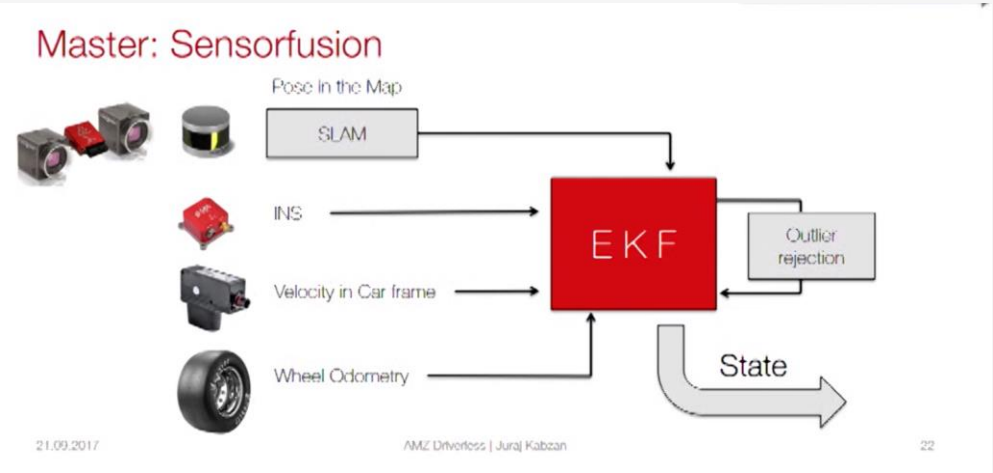
- Point cloud pre-processing
  - Ground removal, etc.
- Cone segmentation and clustering
- Local cone map creation
- “Fast SLAM” used for mapping
- Note that problem is much simpler,

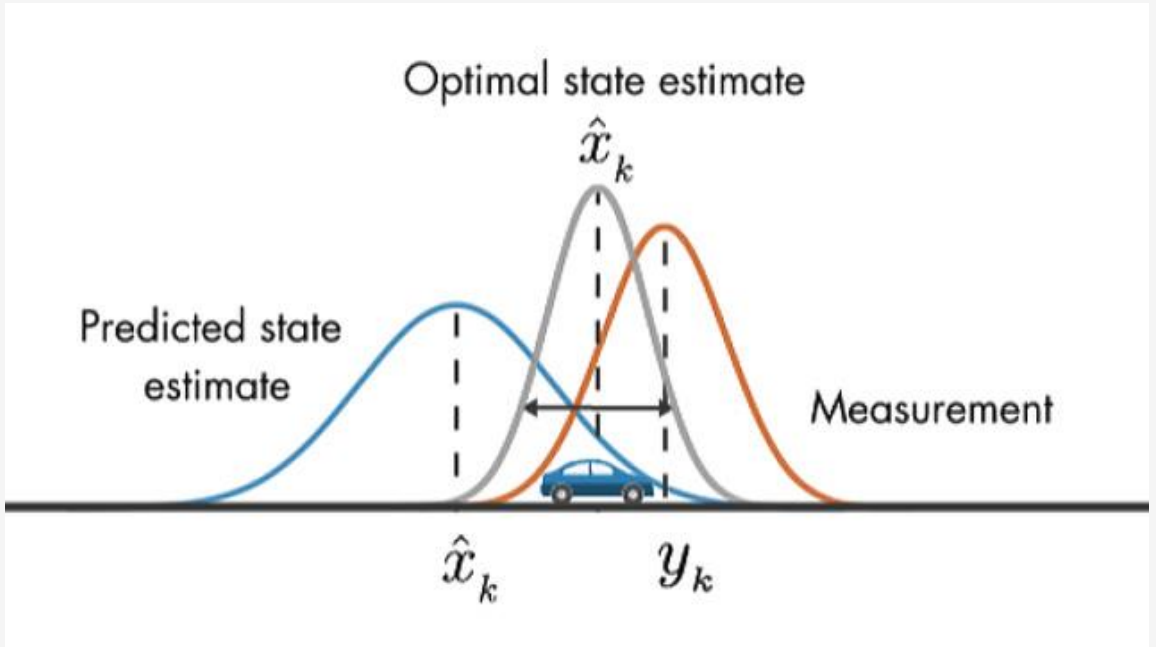
  Basically “Fusing” multiple maps



EKF for Data Fusion

- Extended Kalman Filter (extended means non-linear)
  - Used to fuse multiple sources of data
  - Ensures robustness, accuracy and real-time latency





Vehicle dynamics control

- Two modes
  - Discovery mode:
    - Active during the first loop, used to build the map, slow
  - Race mode:
    - Map must be available
    - Model Predictive Control with dynamic vehicle model
    - Source code available

Conclusions

- Visual SLAM is not necessary

  - but used as a back-up to ensure robustness

- EKF and MPCC quite difficult

  - EKF and data fusion could be a separate project

  - MPCC could be a separate project

- Large part of the project is open-sourced

  - we'll use that as a starting point

Back-up: SLAM

- **SLAM** stands for **S**imultaneous **L**ocalization **A**nd **M**apping
  - mapping = building map of the environment
  - localization = calculating robot's position in the map
  - map = point cloud of landmarks (not necessarily cones)
  - loop closure = adjusting the map to match loop's end to the beginning
  - localization mode = "freezing" the map, estimating only the position

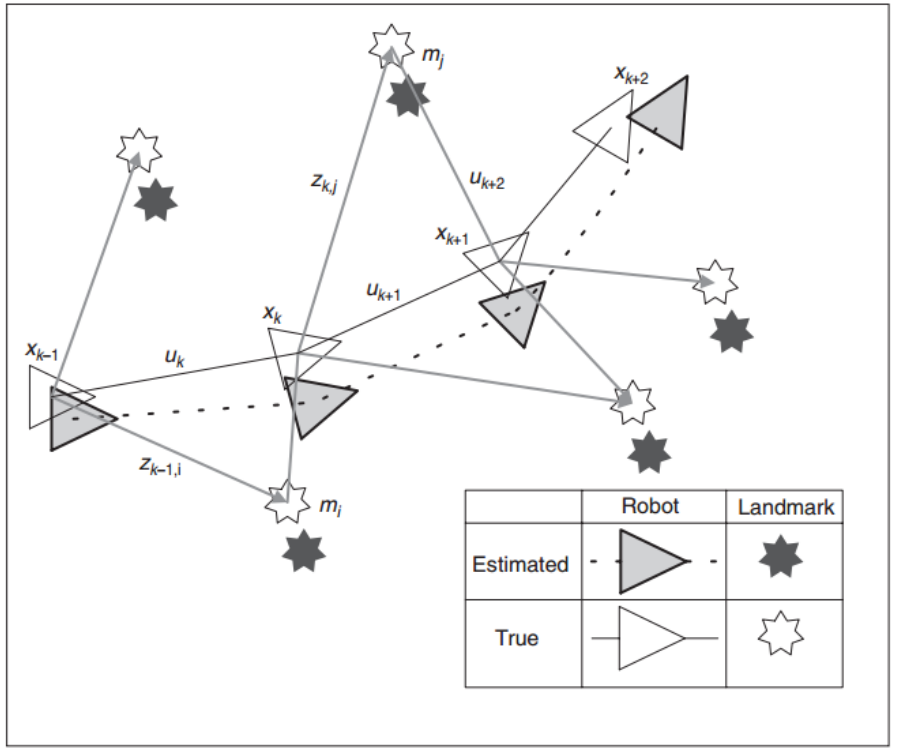


Figure 1. *The essential SLAM problem. A simultaneous estimate of both robot and landmark locations is required. The true locations are never known or measured directly. Observations are made between true robot and landmark locations.*

VISL Vision and Image Sciences Laboratory

Back-up: MPCC

- MPC: simple example

e.g.:

$$\boldsymbol{x} = [x, v] \quad \boldsymbol{u} = [a]$$

$$x_{t+1} = x_t + v_t \cdot dt + a_t \cdot 0.5dt^2$$

$$v_{t+1} = v_t + a_t \cdot dt$$

$$\rightarrow A = \begin{pmatrix} 1 & dt \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0.5dt^2 \\ dt \end{pmatrix}$$

$$\boldsymbol{J_t} = \left(x_t - x_{planned}\right)^2$$

Linear time-invariant convex optimal control

$$\begin{array}{ll} \text{minimize} & J = \sum_{t=0}^{\infty} \ell(x(t), u(t)) \\ \text{subject to} & u(t) \in \mathcal{U}, \quad x(t) \in \mathcal{X}, \quad t = 0, 1, \ldots \\ & x(t+1) = Ax(t) + Bu(t), \quad t = 0, 1, \ldots \\ & x(0) = z. \end{array}$$

- variables: state and input trajectories $x(0), x(1), \ldots \in \mathbf{R}^n$, $u(0), u(1), \ldots \in \mathbf{R}^m$
- problem data:
  - dynamics and input matrices $A \in \mathbf{R}^{n \times n}$, $B \in \mathbf{R}^{n \times m}$
  - convex stage cost function $\ell : \mathbf{R}^n \times \mathbf{R}^m \rightarrow \mathbf{R}$, $\ell(0, 0) = 0$
  - convex state and input constraint sets $\mathcal{X}$, $\mathcal{U}$, with $0 \in \mathcal{X}$, $0 \in \mathcal{U}$
  - initial state $z \in \mathcal{X}$

Back-up: Software

- Good software infrastructure framework is crucial for the project

  ROS used extensively

- Open source:

  VSLAM with data

  LIDAR data and more data

  ROS simulator (gazebo)

  MPCC

  ROS System skeleton

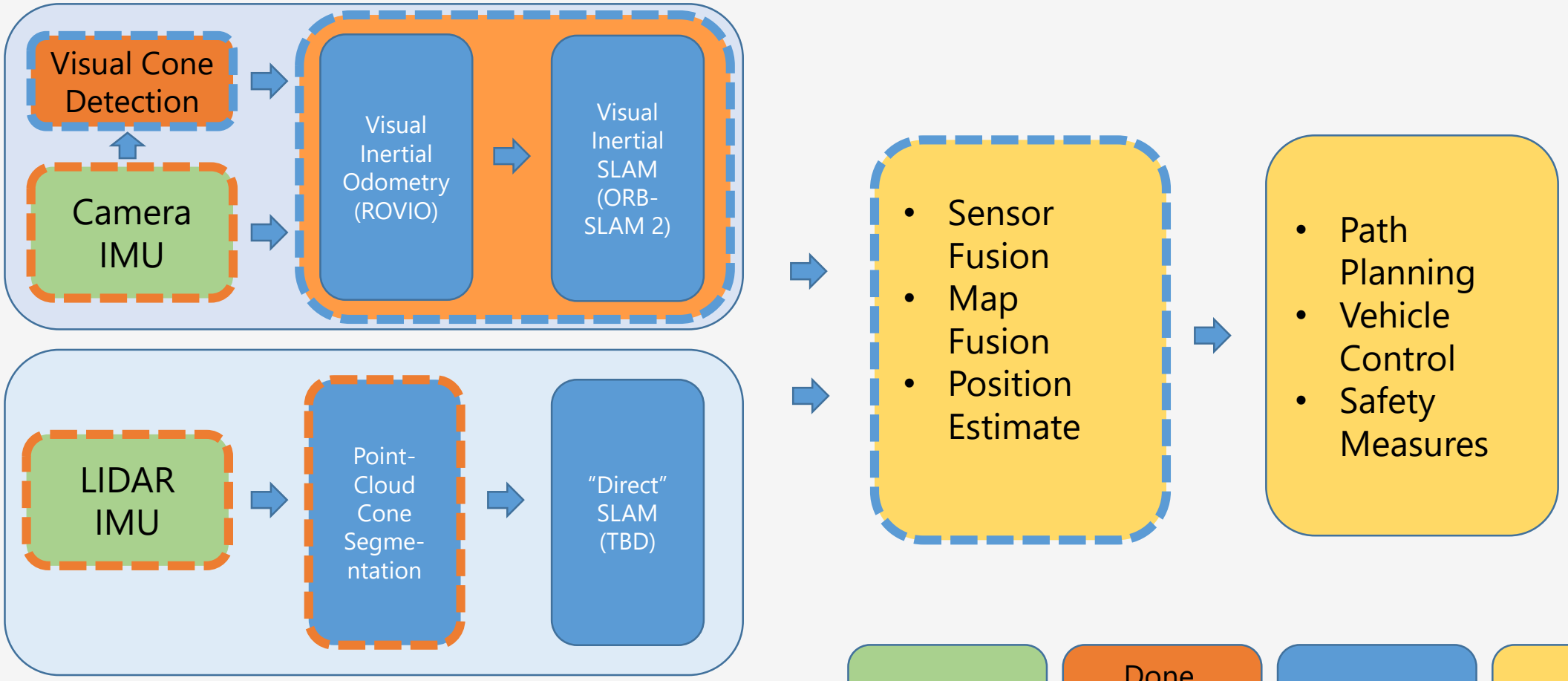
Back-up: Relevant papers of AMZ Driverless

- Talks: 1, 2, 3, 4
- Design of an Autonomous Racecar: Perception, State Estimation and System Integration
- Redundant Perception and State Estimation for Reliable Autonomous Racing
- Learning a CNN-based End-to-End Controller for a Formula Racecar
- Path following control for autonomous formula racecar: Autonomous formula student competition
- Design of an Autonomous Race Car for the Formula Student Driverless (FSD)

Our Project

- Main goal is to build an algorithm to reconstruct the race track during the first loop and provide an estimate of the vehicle's position on the map while driving (SLAM)
- Since the project plays central role in autonomous navigation and perception it will also define the shape of the whole system
- Thus, we also need to ensure that SW/HW infrastructure is suitable for a competitive solution in the future

System Overview

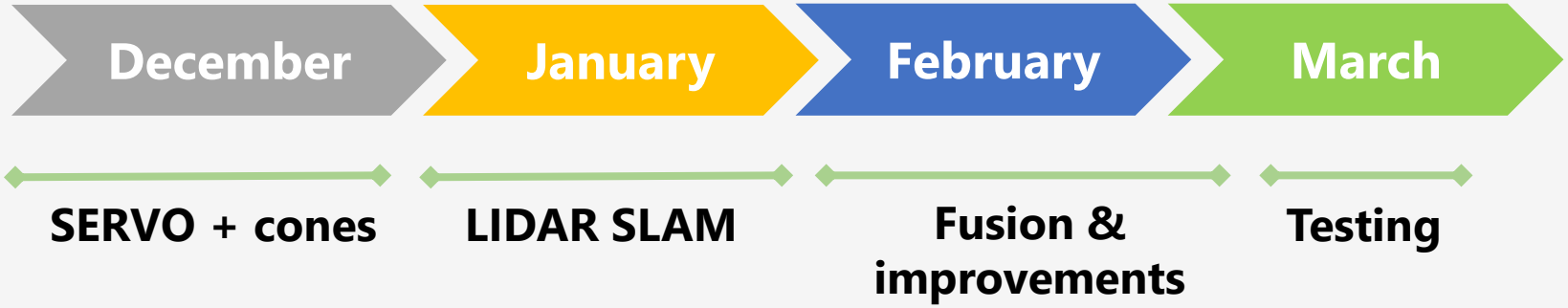


VISL Vision and Image Sciences Laboratory

Project Timeline Considerations

- Limited by data and equipment availability
  - need a ZED camera (good infrastructure, built-in IMU) with steady cam / stand
  - waiting for LIDAR
  - then we'll need to capture content and make use of it
- Adjust roadmap accordingly
  - while no data available work on theory and open-source
  - start by testing SERVO on provided data, then test with ours
  - when LIDAR available, start working on segmentation and mapping

Project Timeline



SERVO + cones

- SERVO code open-sourced
- Need to validate and integrate
- Later, add cones explicitly

LIDAR SLAM

- Currently looking for data
- First will need to segment cones
- Then we'll look for an algorithm

Fusion & improvements

- Add maps fusion
- Improve the system as a whole

Testing

- Build a Gazebo simulator
- Test in real-world environment
- Document the project

VISL

Vision and Image Sciences Laboratory

Project Goals

- Learn about the field, read prior works
- Design a skeleton for autonomous navigation system
- Design, implement and test relevant algorithms (based on existing solutions)
- Validate the system (consider building an end-to-end simulator)
- Document the work process
- Test on real vehicles
- Demonstrate & present the final work

Main Challenges / Open Questions

- Data and equipment availability
  - Need (preferably stereo) camera with IMU, LIDAR, all calibrated and synchronized
  - Need a couple of sequences in close to real-world scenarios (first loop is slow, next 9 - fast, real cones,..)
- Previous work should be finalized and validated
  - Cone detection – speed/accuracy?
  - LIDAR synchronization and calibration
- Formal system definition
  - Analyze available options and agree on Roadmap
  - Agree on SW platform (we tend to ROS: most appropriate but has steep learning curve)
  - Our task: choose missing components (“direct” SLAM, cone segmentation, etc.)

Good Luck

Vision and Image
Sciences Laboratory