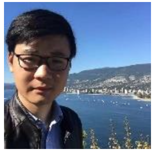
Motion Planning for Mobile Robots



主讲人 Fei Gao

Ph.D. in Robotics
Hong Kong University of Science and Technology
Assistant Professor，Zhejiang University

Outline

1. Introduction
2. Course Outline
3. Typical Planning Methods Overview
4. Map Representation
5. Pre-requirement
6. Homework

Introduction

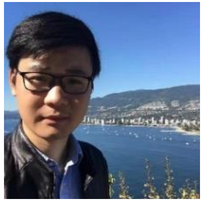
About this Course

- This course is about:
  - Academism (old school) planning pipeline
  - Path finding algorithm
  - Trajectory generation/optimization
  - Real-time software implementation
- This course is NOT about:
  - Dynamics Modelling ☹
  - Robot design ☹
  - End-to-end navigation ☹
  - Learning-based method ☹
  - Mathematical/theoretical proof ☹
- This course may also cover:
  - Autonomy for mobile robots
  - Paper recommendations
  - Cutting-edge research direction/ethics

Basic Expectation

- Discipline:
  - Every researcher has his own taste/style. Since you choose this course, please follow my style.
  - Q&A only at appointed office time.
  - Finish your homework by yourself.
- Project:
  - Basic algorithm validation (MATLAB)
  - Sophisticated engineering implementation (ROS/C++)

Teaching Team



Instructor 1: **Fei Gao**

Assistant Professor, ZJU

Ph.D, HKUST

Email: fgaoaa@connect.ust.hk

Instructor 2: **Chaoqun Wang**

Research Fellow, CUHK

Ph.D, CUHK

Email: zychaoqun@gmail.com

Instructor 3: **Shupeng Lai**

Research Fellow, NUS

Ph.D, NUS

Email: shupenglai@gmail.com

Instructor 4: **Delong Zhu**

Ph.D candidate, CUHK

Email: dlzhu@ee.cuhk.edu.hk

Information

- Lecture time:
  - Every Friday night 19:00, except October 4th

- Course Community:
  - www.shenlanxueyuan.com/course/188/threads/show

- Course Website:
  - www.shenlanxueyuan.com/course/188

- Course Wechat Group:
  - Connect the class teacher(Wechat: shenlancas)

Workload

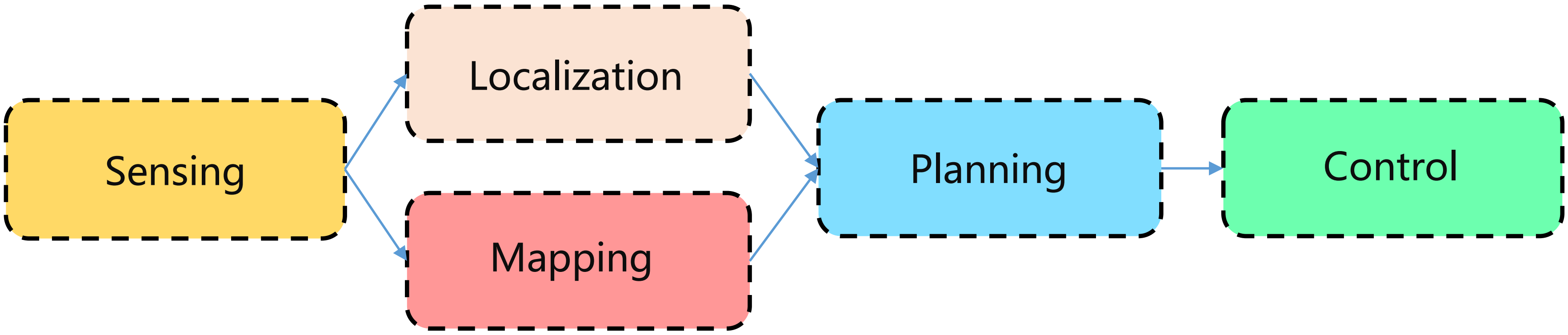
- Expected Student Background:
  - Linear algebra
  - Probability
  - MATLAB programming skills
  - C++ programming skills (VERY IMPORTANT)
  - Linux
  - Love robots ☺
- Workload:
  - Attend lectures
  - Lots of project work
  - Have fun with robots ☺

What is autonomous robot

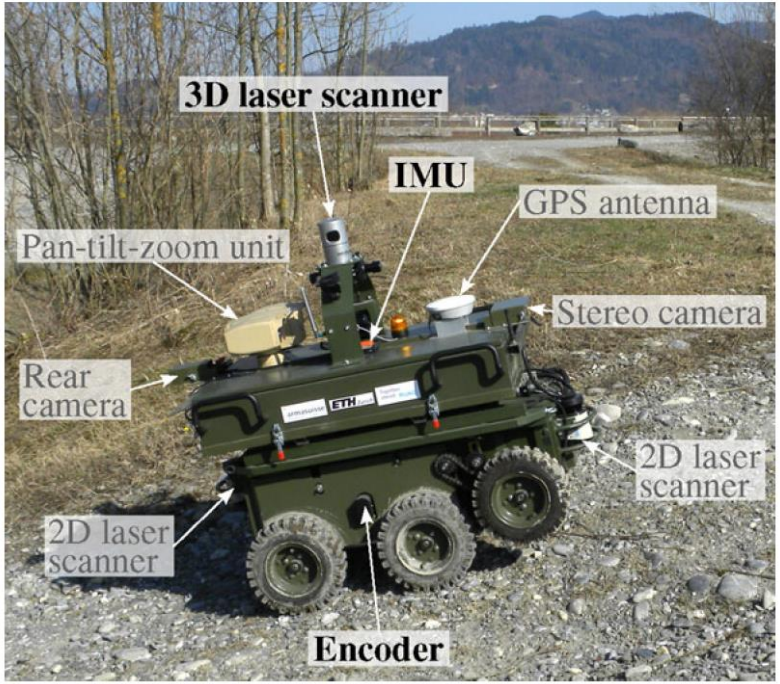
Definition: an autonomous robot is a robot that performs behaviors or tasks with a high degree of autonomy (without external influence).

Subjects: Computer Science, Automation, Mechanism, Electronics …

- Perception-Planning-Control action loop

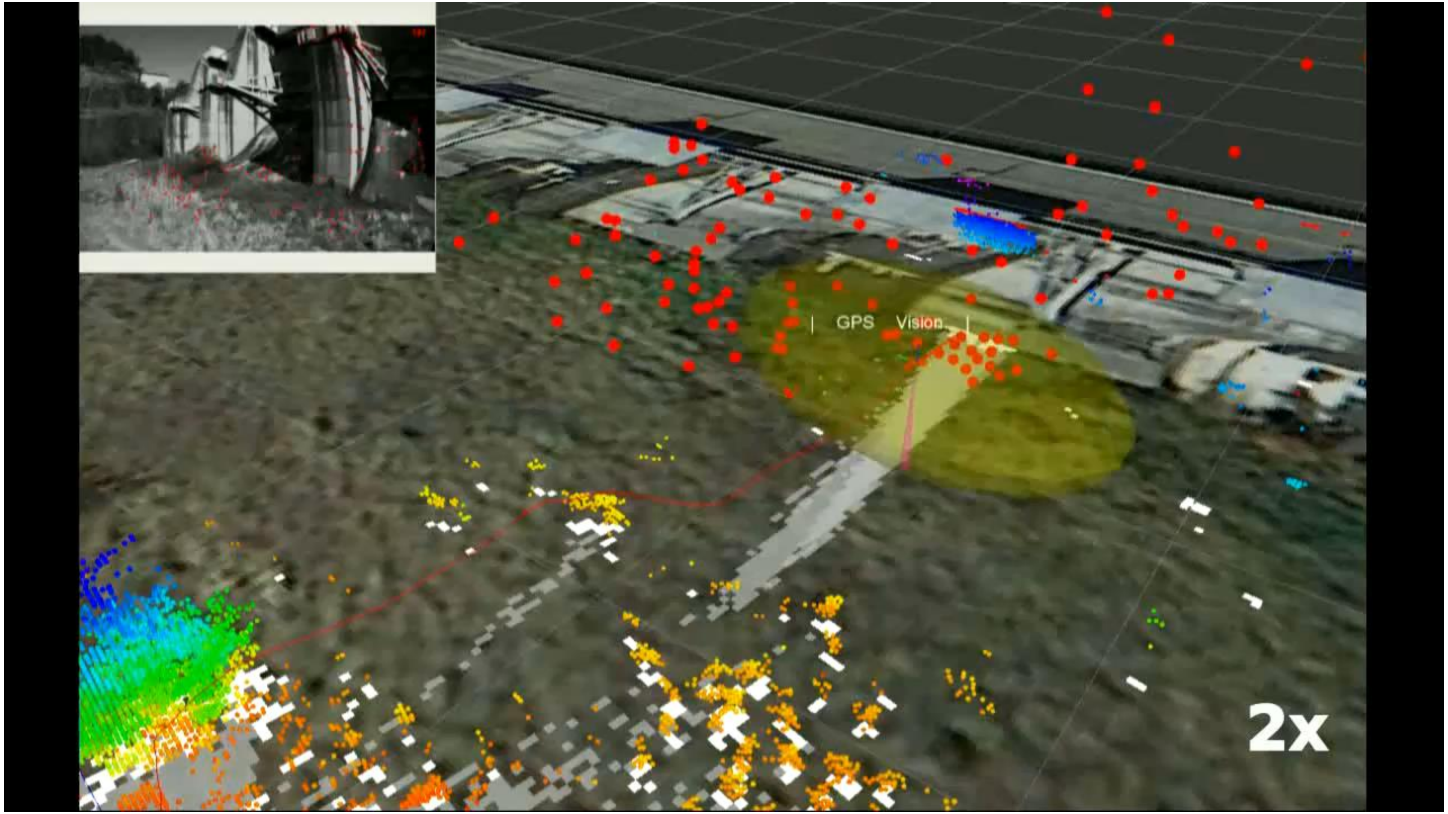


Autonomous robot: applications



Aerial Robots

Dam Inspection



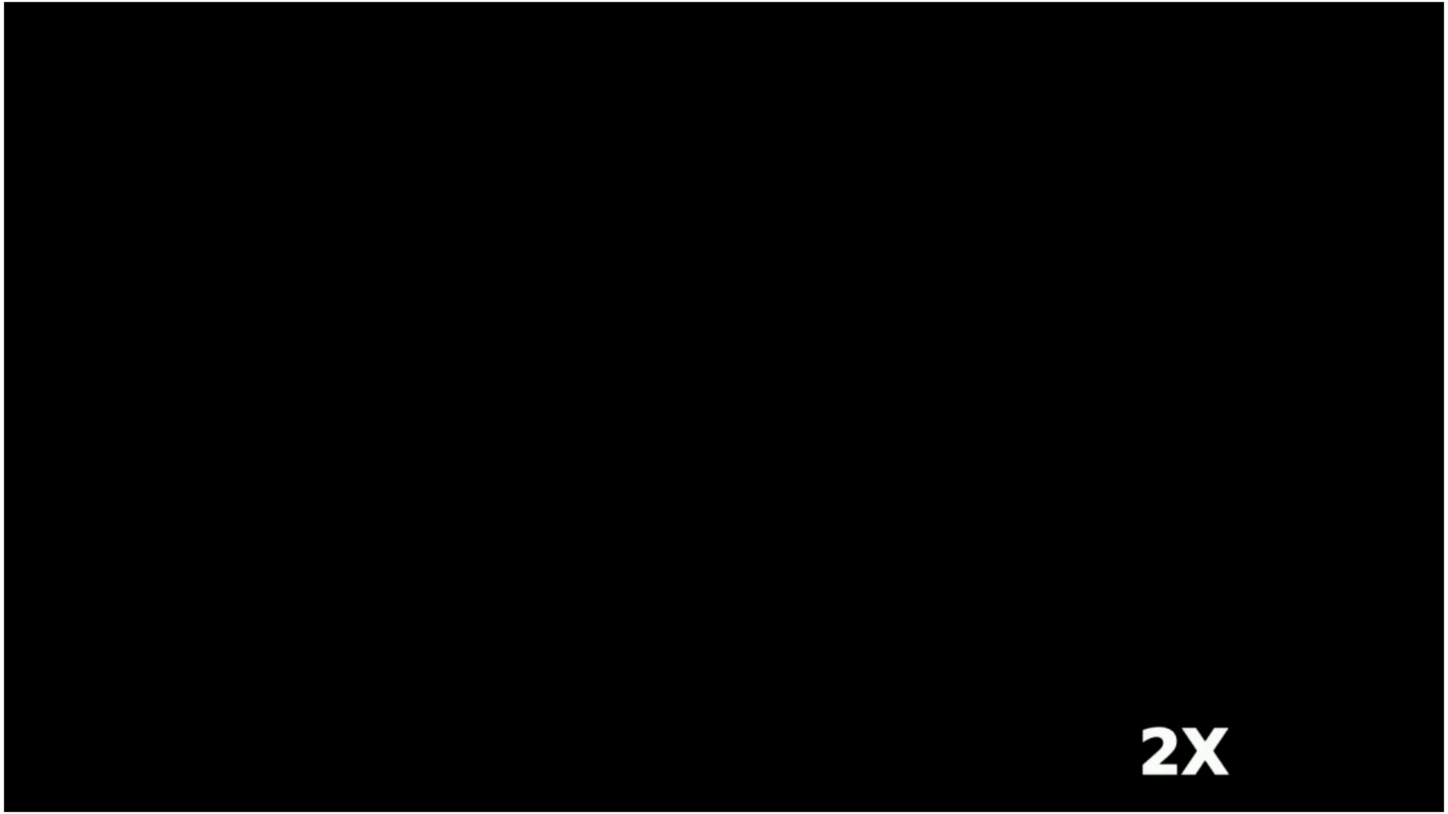
Shen, et al, 2014

Dam Inspection

Ozaslan, et al, 2014

Cellular Tower Inspection

Precision Farming



Shen, et al, 2014

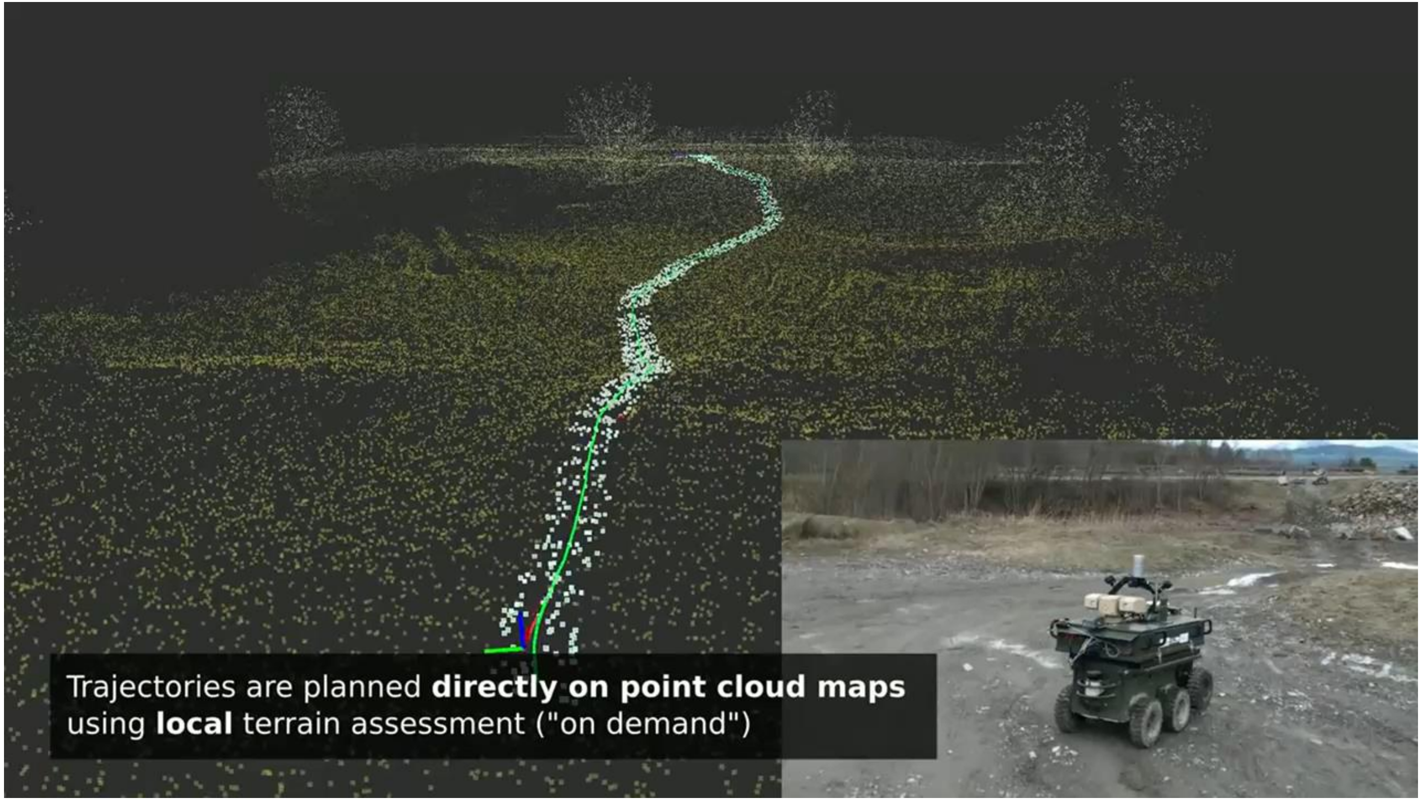
Delivery

Ground Robots

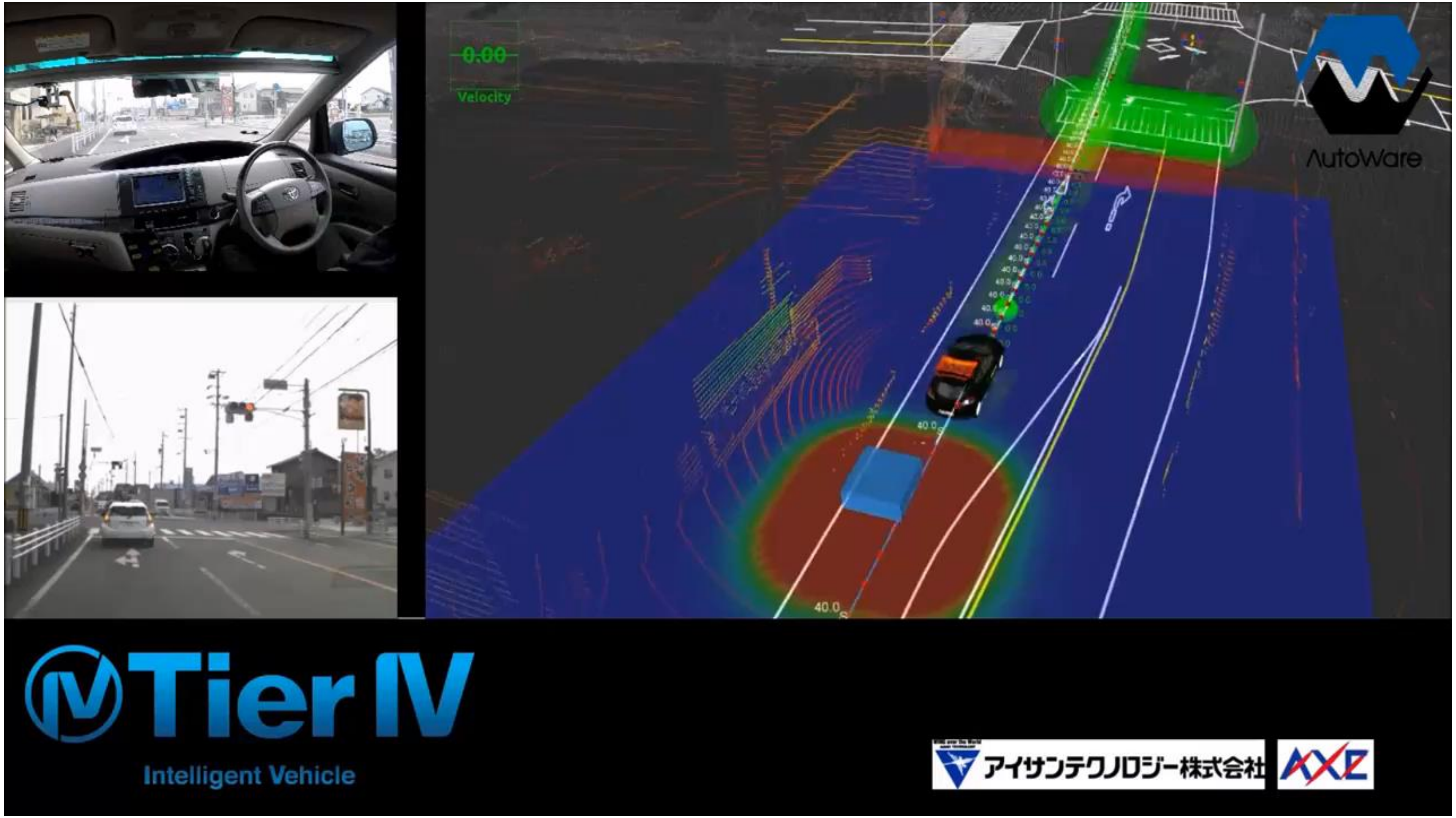
Search and Rescue

Housekeeping

Hills Driving



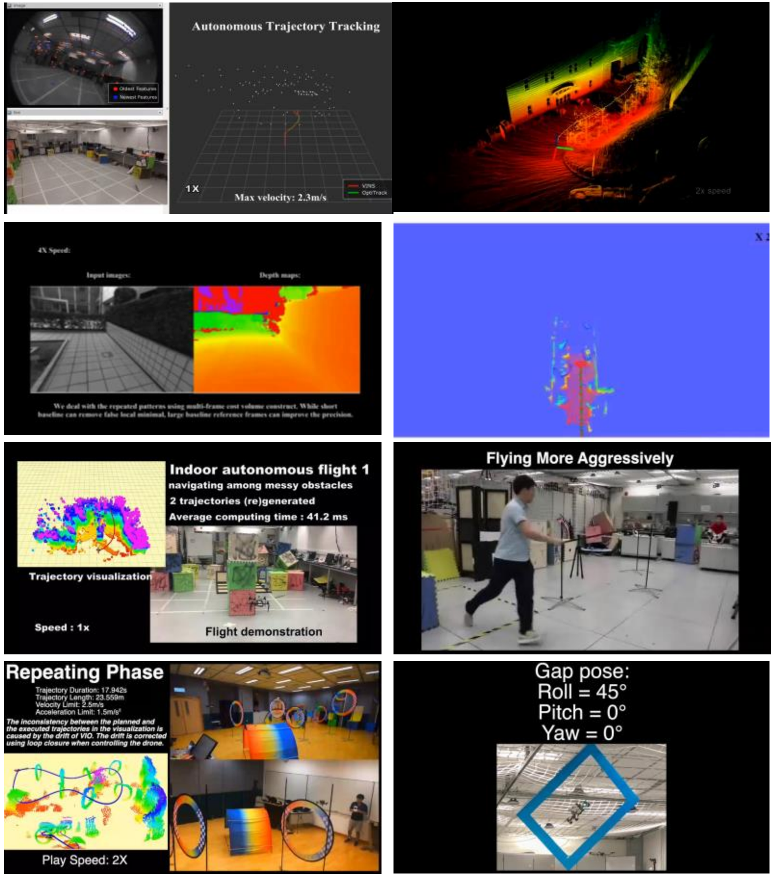
Urban Driving



Crowd Driving

What is autonomous robot

- Estimation
  - Low latency
  - High accuracy & consistency
- Perception
  - 3D sensing & dense perception
  - Map fusion & integration for planning
- Planning
  - Complex & unknown environments
  - Safety & dynamical feasibility
  - Limited sensing & computation
- Control
  - Aggressive maneuvers
  - Smooth trajectory tracking



What is motion planning

- Basic requirements
  - Safety: collision avoidance
  - Smoothness: energy saving, comfort
  - Kinodynamic feasibility: executable, controllable
- Old-school pipeline
  - Front-end path finding
    - Search for an initial safe path
    - Low dimensional
    - Discrete space
  - Back-end trajectory generation
    - Search for an executable trajectory
    - High dimensional
    - Continuous space

How to do robotics research

- Find a problem
  - ✓ A problem only in your imagination is not a problem at all.
  - ✓ A robotician must be an engineer first.
  - ✓ Hot topic is just hot.
  - ✓ Be honest.
- Solve a problem
  - ✓ Don’t intentionally making things complicated.
  - ✓ Simple but effective solution is always preferable.
  - ✓ Simulation tells nothing, show everyone real robots.
  - ✓ Solve real problems.
  - ✓ Solve it 100%, or not.

How to do motion planning

- Overall knowledge of planning
  - ✓ Choose suitable methods for different scenarios.
  - ✓ Design customized strategy.
- Dirty hands
  - ✓ Don't wait, don't just read papers. Do it yourself.
  - ✓ A lot of coding work.
  - ✓ A lot of field work.
- Know the whole system well
  - ✓ Take care every component in your robot.

Groups and Researchers

University of Pennsylvania

- GRASP Lab,Vijay Kumar
  Research Interests: planning, control, swarm
  Homepage: www.kumarrobotics.org

Massachusetts Institute of Technology

- Jonathan How
  Research Interests: modelling, control, planning
  Homepage: www.mit.edu/~jhow

- Nicholas Roy
  Research Interests: perception, learning
  Homepage: groups.csail.mit.edu/rrg

Carnegie Mellon University

- Nathan Michael
  Homepage: www.rislab.org

- Sebastian Scherer
  Research Interests: perception, planning
  Homepage: theairlab.org

University of California, Berkeley

- Markus Mueller
  Research Interests: control, planning

ETH Zurich

- ASL Team, Roland Siegwart
  Research Interests: perception, control
  Homepage: asl.ethz.ch

- Raffaello D'Andrea
  Research Interests : control, swarm
  Homepage: raffaello.name

University of Zurich

- Davide Scaramuzza
  Research Interests : perception, control
  Homepage: rpg.ifi.uzh.ch

Hong Kong University of Science Technology

- Shaojie Shen
  Research Interests : UAV
  Homepage: uav.ust.hk

- Ming Liu
  Research Interests : UGV
  Homepage: ram-lab.com

Young researchers who most related to me

Helen Oleynikova

helenol.github.io

Graduate at Jan. 2019, from ASL team, ETH

Supervised by Roland Siegwart

Author of VoxBlox

Planning and mapping

Sikang Liu

github.com/sikang

Graduate at Aug. 2018, from GRASP Lab,UPenn

Supervised by Vijay Kumar

Planning

Course Outline

Front-end: Path finding

- SEARCH-BASED PATH FINDING
  - Graph Search Basis
  - Dijkstra and A*
  - Jump Point Search
  - Homework

- SAMPLING-BASED PATH FINDING
  - Probabilistic Road Map
  - Rapidly-exploring Random Tree (RRT)
  - Optimal Sampling-based Methods
  - Advanced Sampling-based Methods
  - Homework

- KINODYNAMIC PATH FINDING
  - Introduction
  - State-state Boundary Value Optimal Control Problem
  - State Lattice Search
  - Kinodynamic RRT*
  - Hybrid A*
  - Homework

Back-end: Trajectory Generation

- MINIMUM SNAP TRAJECTORY GENERATION
  - Differential Flatness
  - Minimum Snap Optimization
  - Closed-form Solution to Minimum Snap
  - Time Allocation
  - Implementation in Practice
  - Homework
- SOFT AND HARD CONSTRAINED TRAJECTORY OPTIMIZATION
  - Soft Constrained Trajectory Optimization
  - Hard Constrained Trajectory Optimization
  - Homework

MDP & MPC

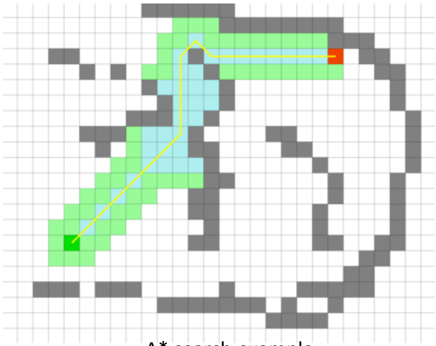
- MARKOV DECISION PROCESS-BASED PLANNING
  - Uncertainties in Planning and MDP
  - Minimax Cost Planning and Expected Cost Minimal Planning
  - Value Iteration and Real-Time Dynamic Programming
  - Homework
- MODEL PREDICTIVE CONTROL FOR ROBOTICS PLANNING
  - Introduction
  - Linear MPC
  - Non-linear MPC
  - Homework

Overview

Front-end: Path Finding

- Sampling-based methods
  - Probabilistic roadmap (PRM)
  - Rapidly exploring random tree (RRT)
  - RRT*, informed RRT*
- Search-based methods
  - General graph search: DFS, BFS
  - Dijkstra and A* search
  - Jump point search

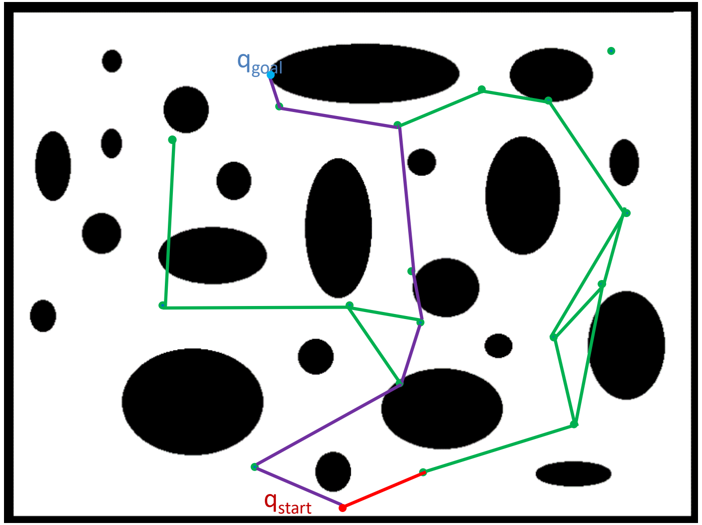
RRT example



A* search example

Sampling-based Method

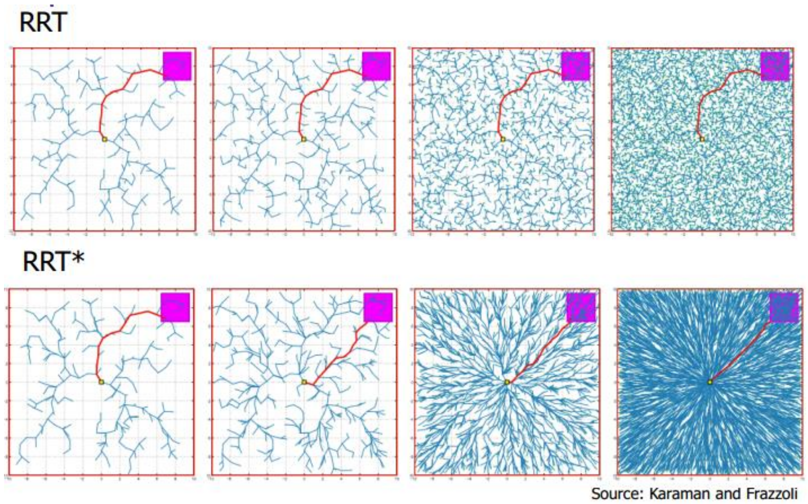
Probabilistic Roadmap (PRM)

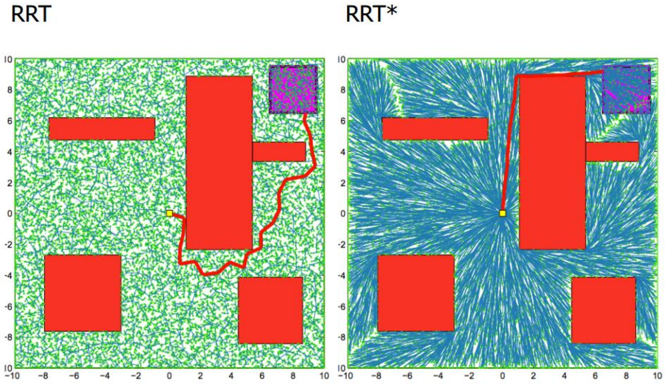


C-space

Sampling-based Method

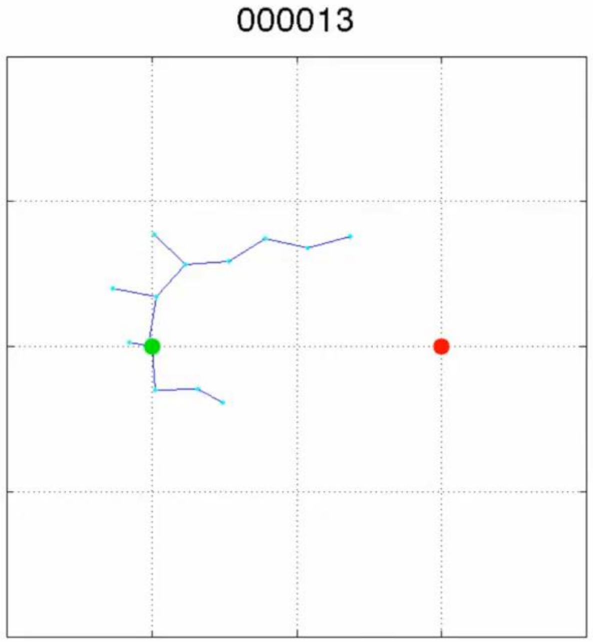
RRT* vs RRT





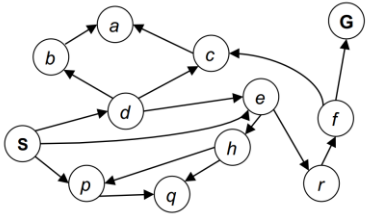
Sampling-based Method

Informed RRT*

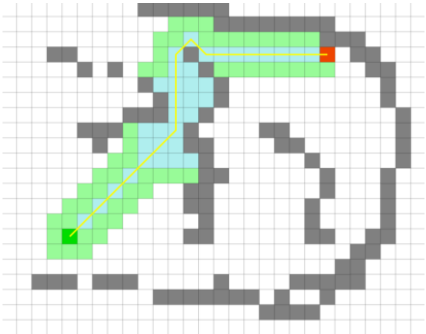


Search-based Method

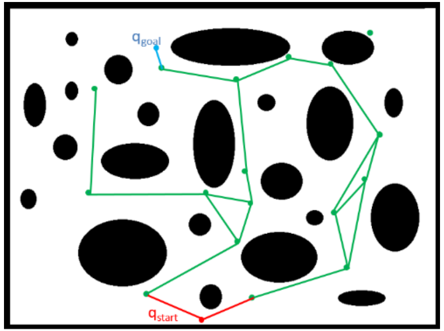
- For every search problem, there's a corresponding state space graph
- Connectivity between nodes in the graph is represented by (directed or undirected) edges



Ridiculously tiny search graph for a tiny search problem



Grid-based graph: use grid as vertices and grid connections as edges

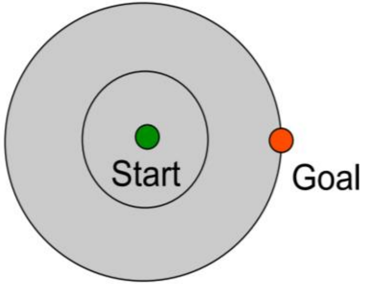


The graph generated by probabilistic roadmap (PRM)

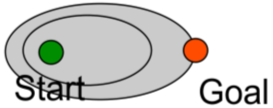
Search-based Method

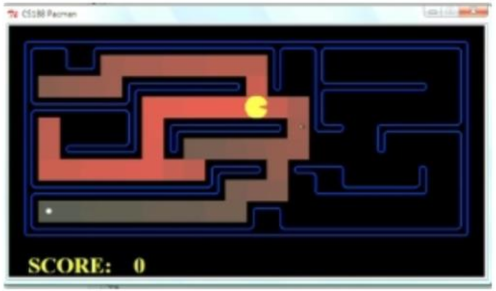
Dijkstra’ s vs. A*

- Dijkstra’s algorithm expanded in all directions



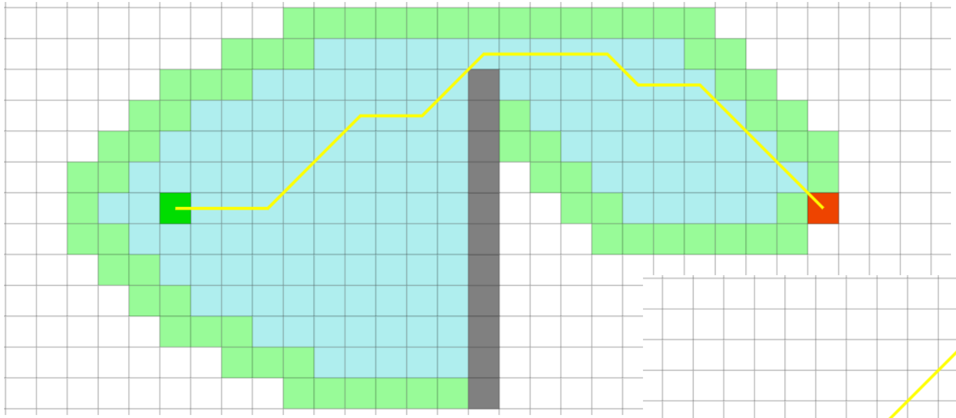
- A* expands mainly towards the goal, but does not hedge its bets to ensure optimality

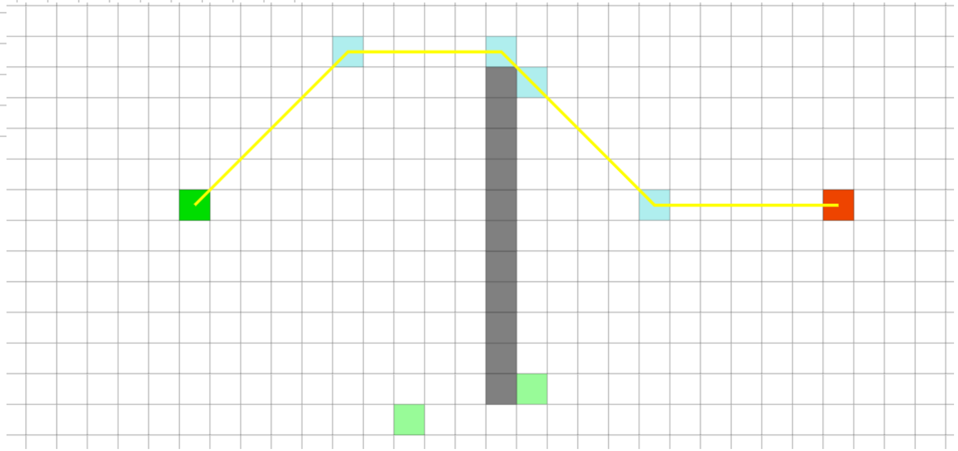




Search-based Method

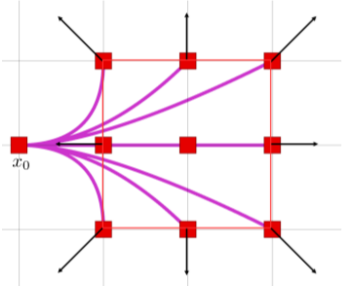
A* vs. JPS

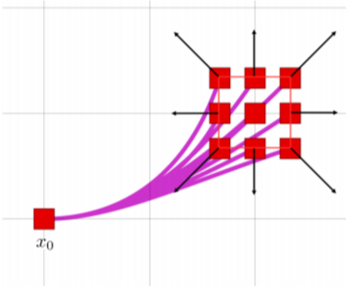


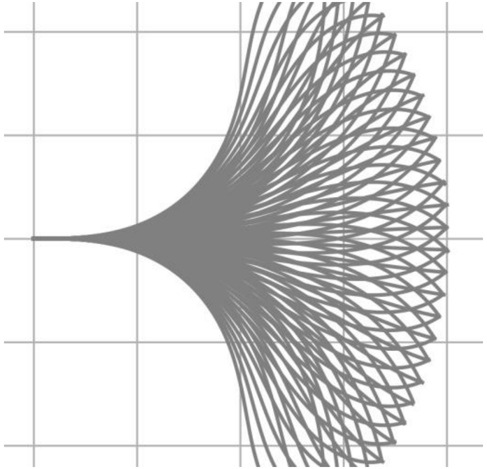


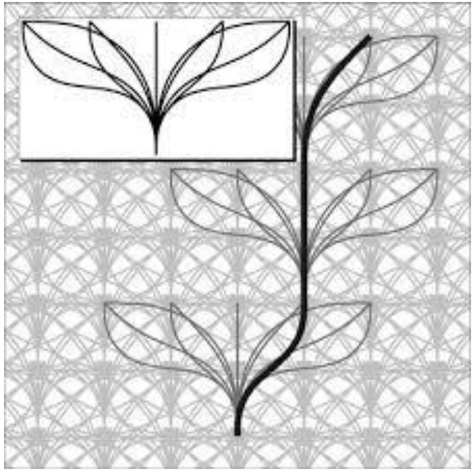
Kinodynamic Path Finding

State Lattice Search



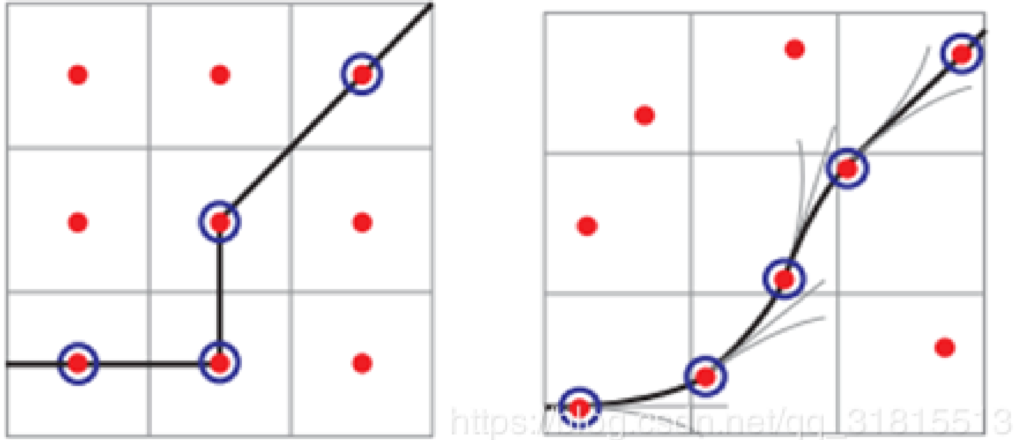






Kinodynamic Path Finding

Hybrid A*



1.Follow A* algorithm

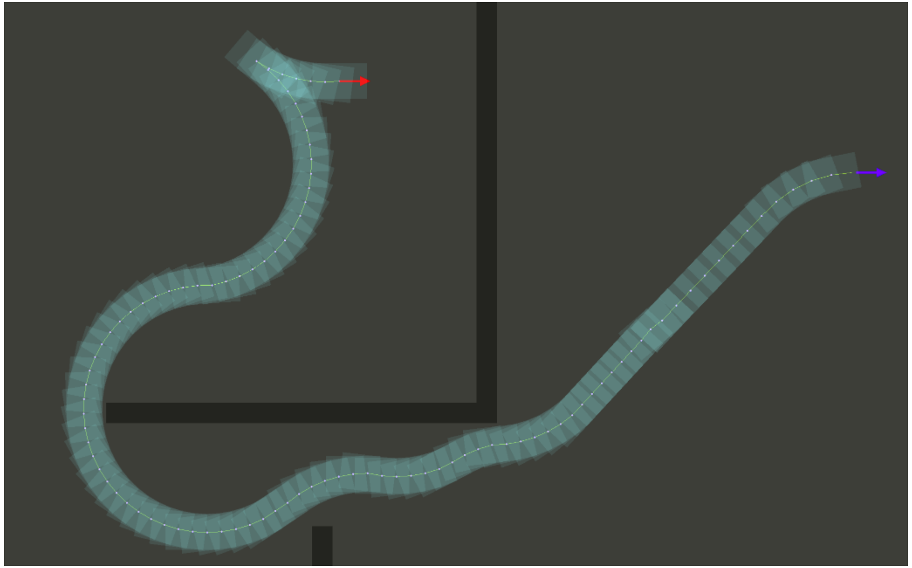
2. Forward simulate states with different discrete control inputs

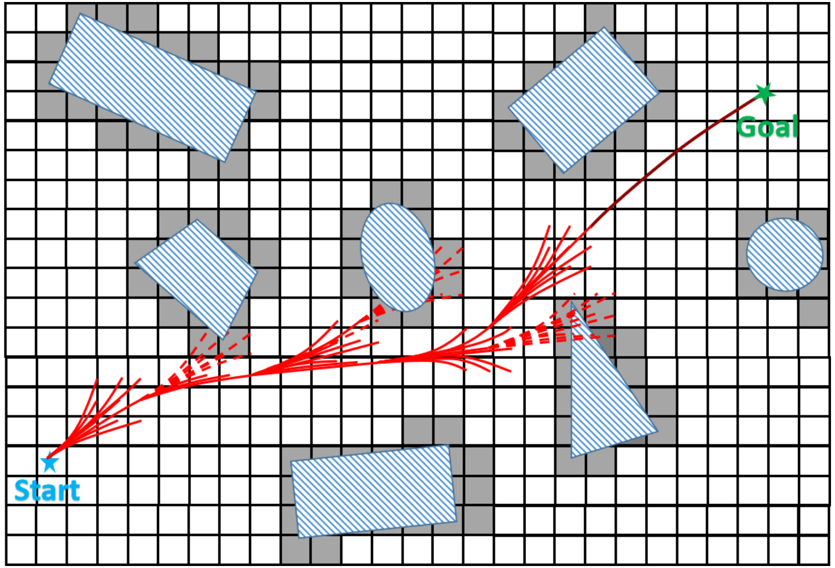
3. Keep only 1 state in each grid

Discrete control

Kinodynamic Path Finding

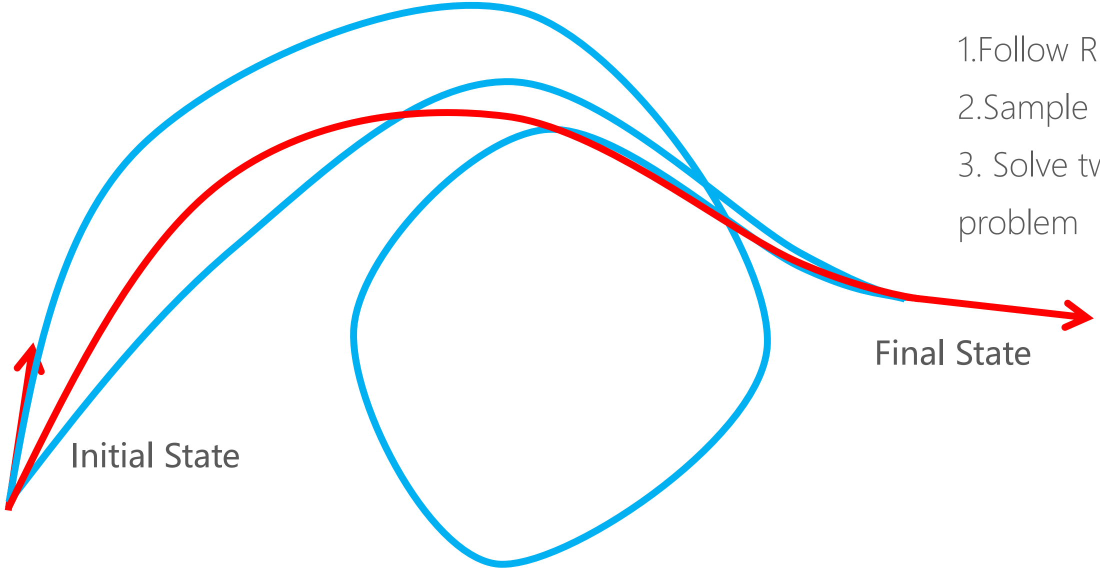
Hybrid A*





Kinodynamic Path Finding

Kinodynamic RRT*



1.Follow RRT* algorithm

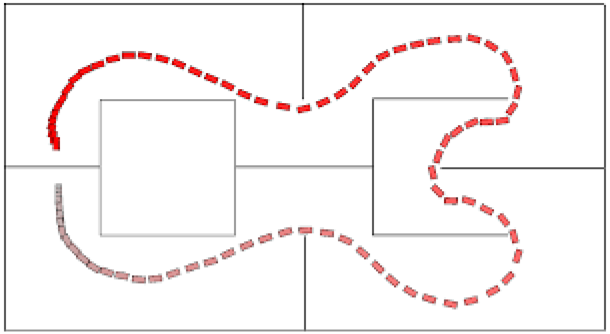
2.Sample a random state

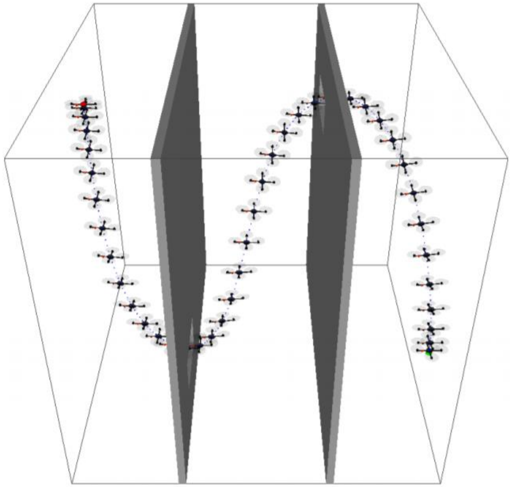
3. Solve two state boundary optimal control problem

Discrete state

Kinodynamic Path Finding

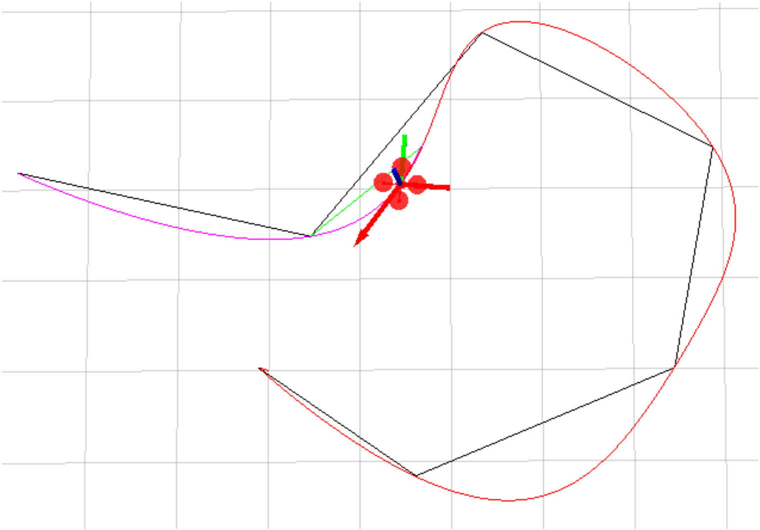
Kinodynamic RRT*

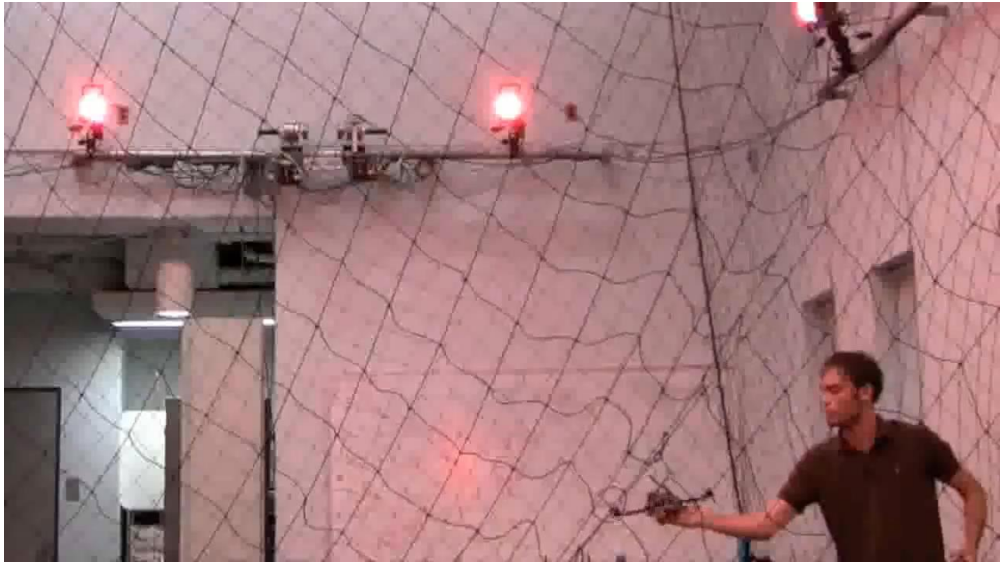




Back-end: Trajectory Optimization

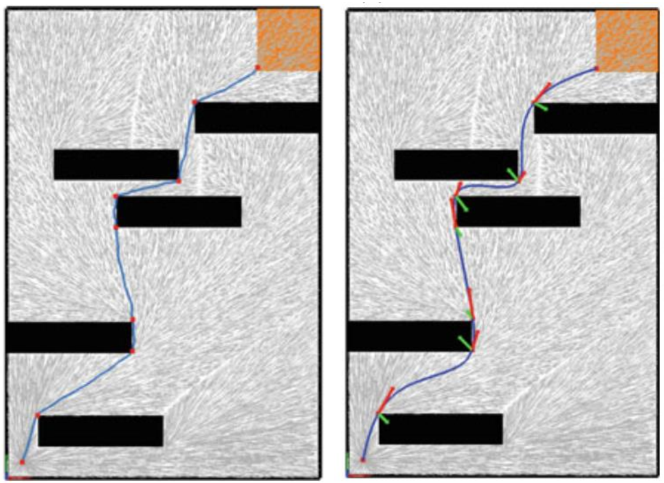
Basic Minimum-snap

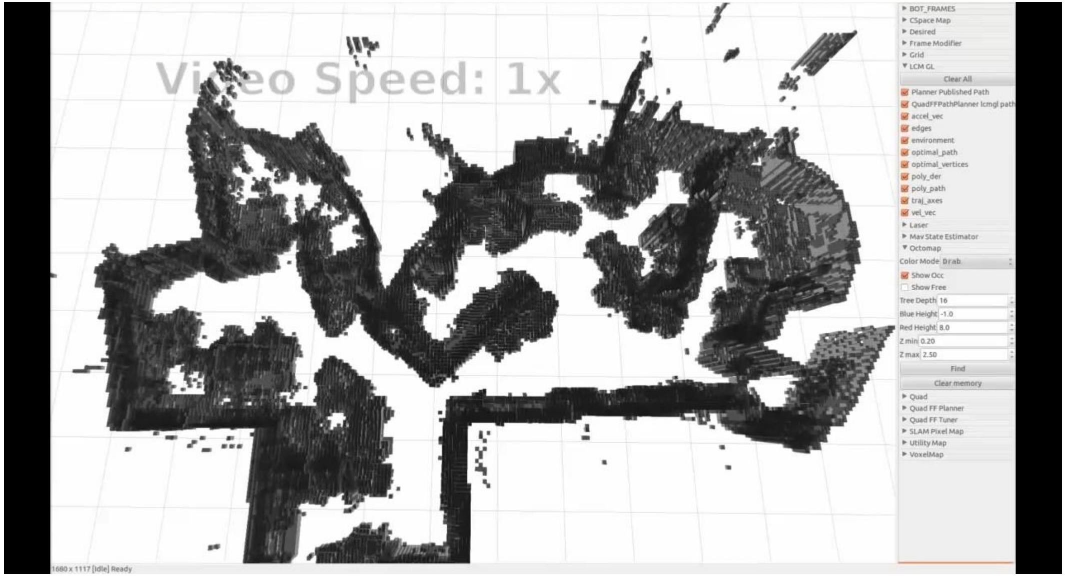




Back-end: Trajectory Optimization

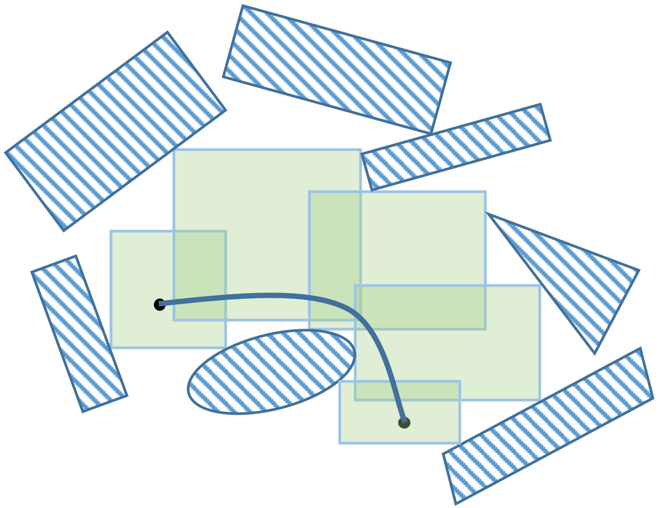
Basic Minimum-snap

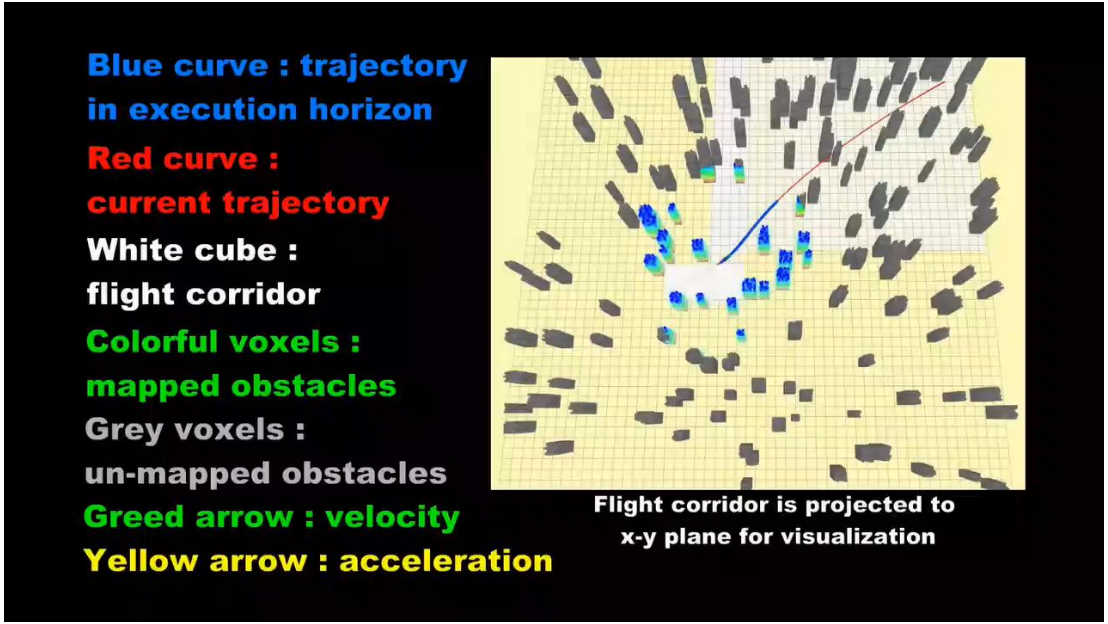




Back-end: Trajectory Optimization

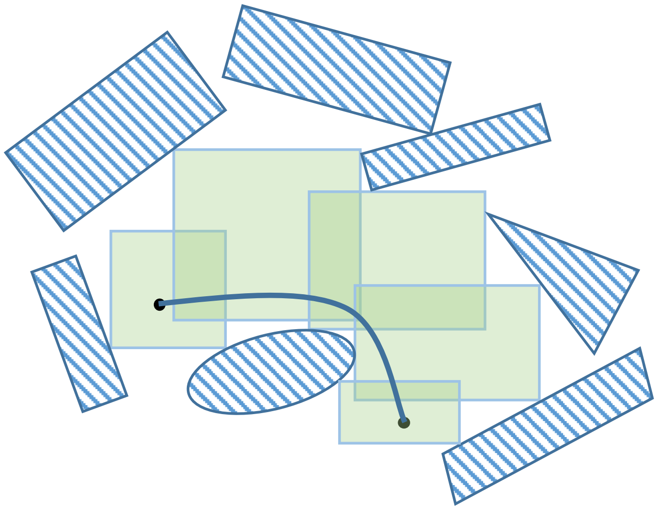
Hard constrained Minimum-snap

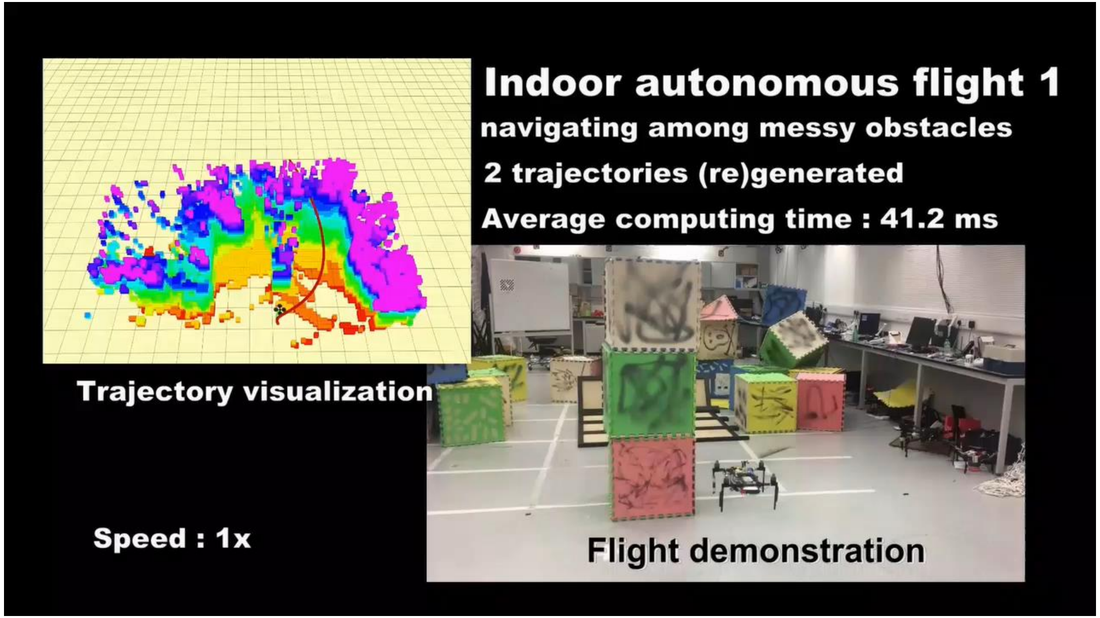




Back-end: Trajectory Optimization

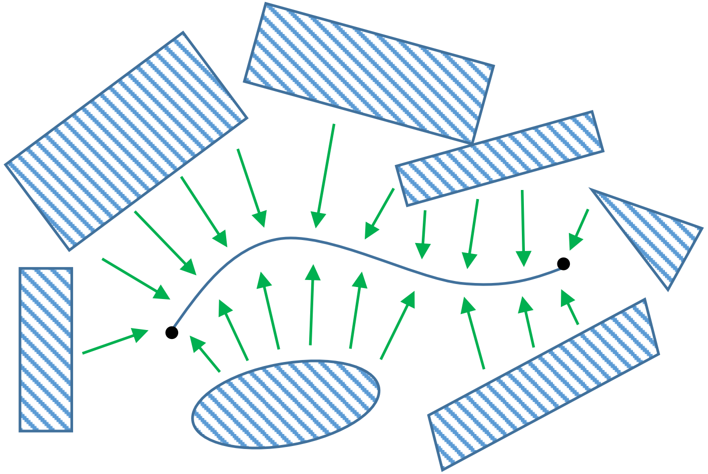
Hard constrained Minimum-snap

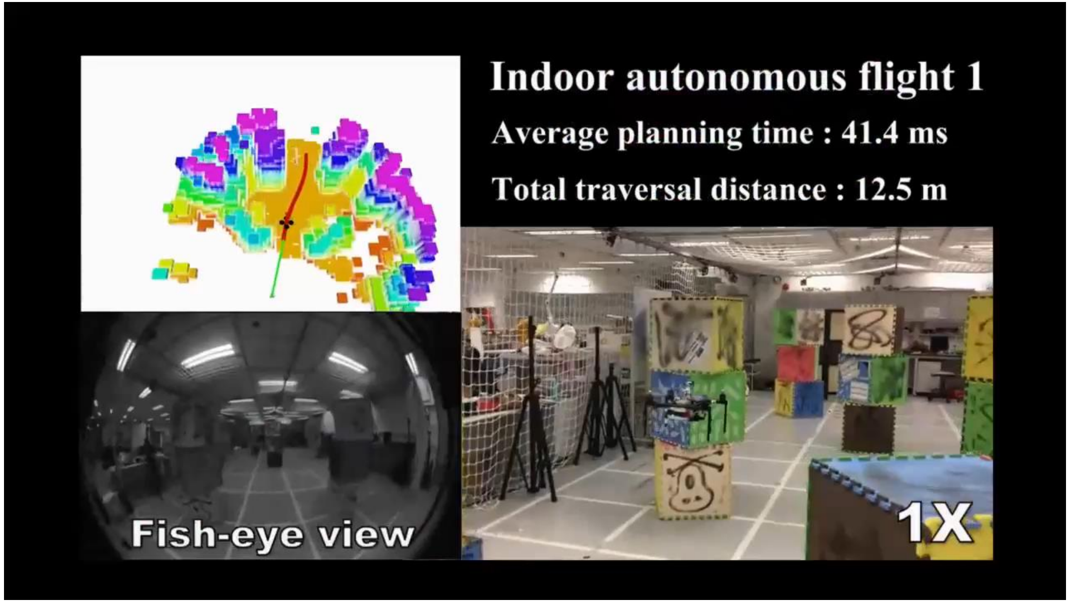




Back-end: Trajectory Optimization

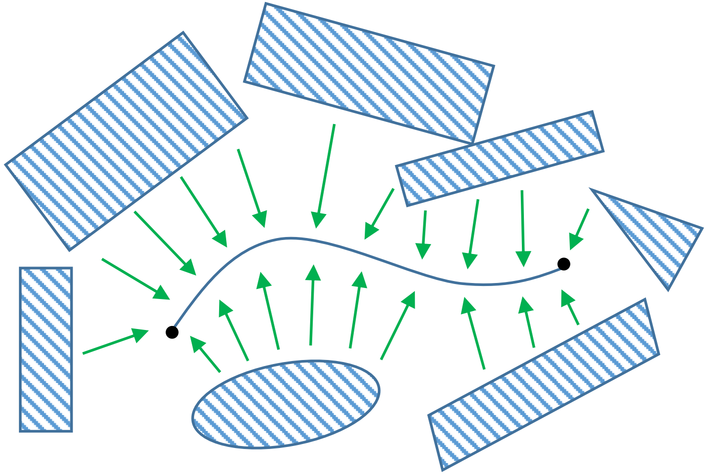
Soft constrained Minimum-snap

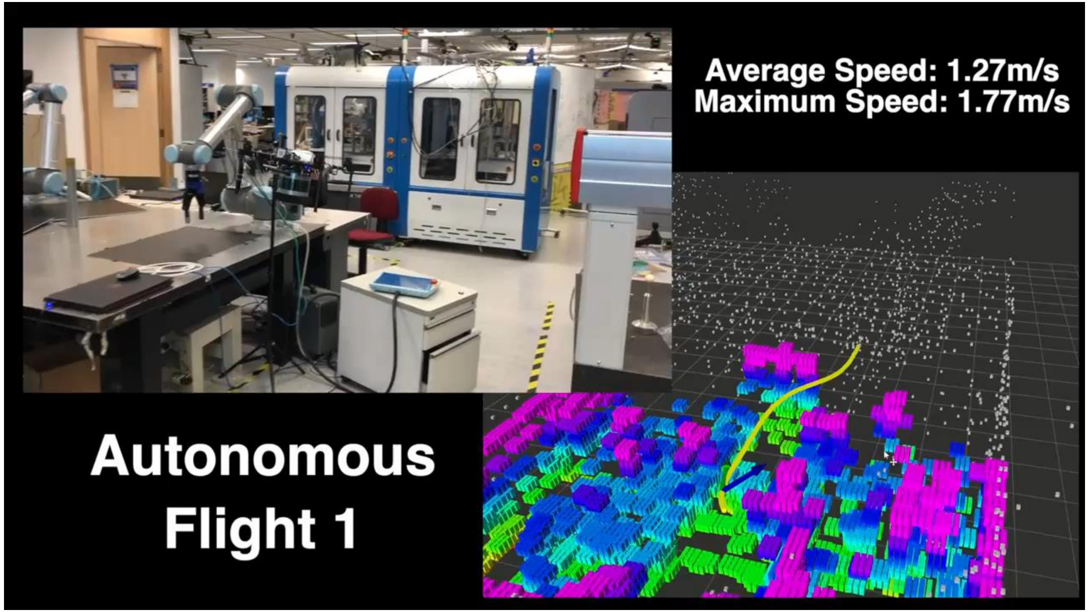




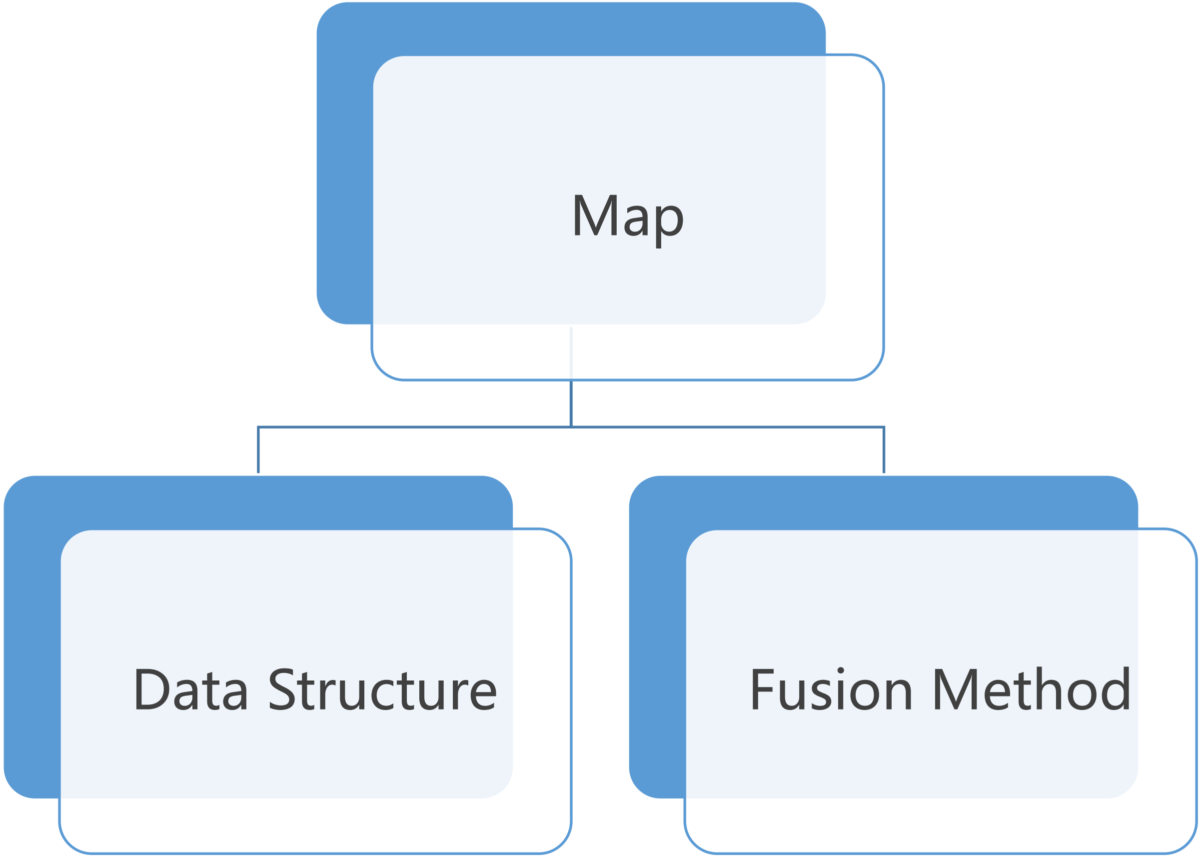
Back-end: Trajectory Optimization

Soft constrained Minimum-snap

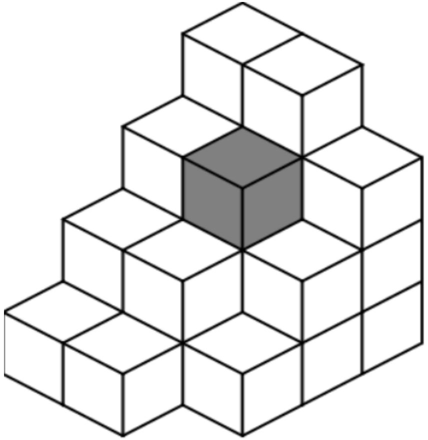


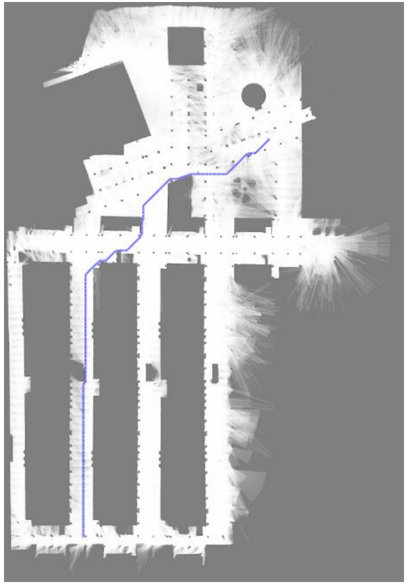


Map Representation

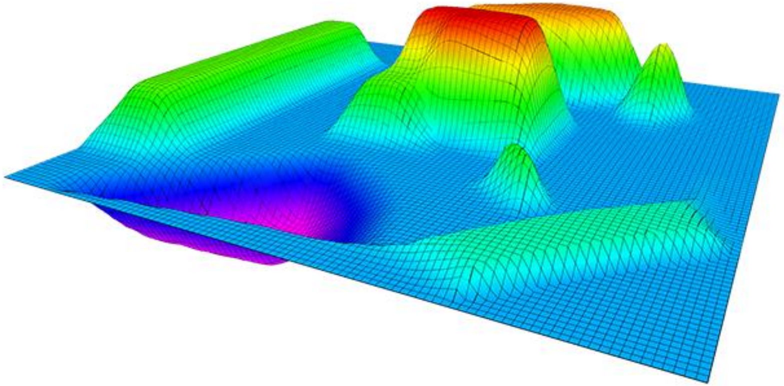

Map
Data Structure
Fusion Method

Occupancy grid map



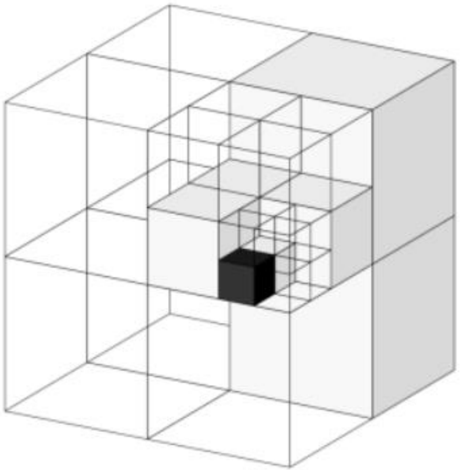


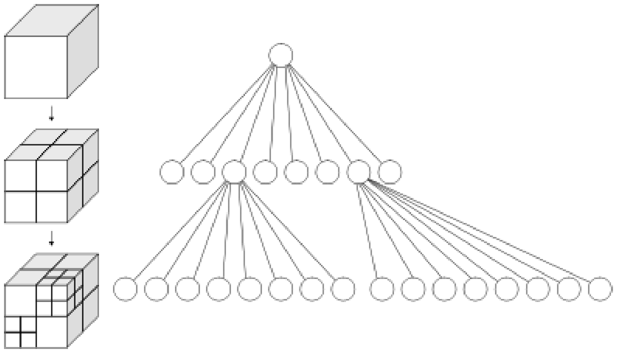
- Most Dense
- Structural
- Direct Index Query



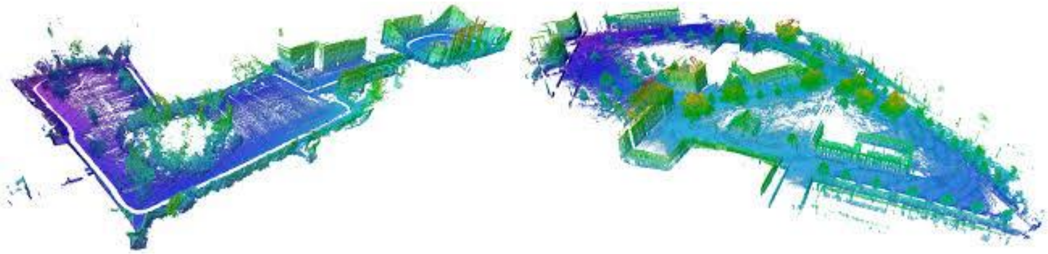
https://github.com/ANYbotics/grid_map

Octo-map



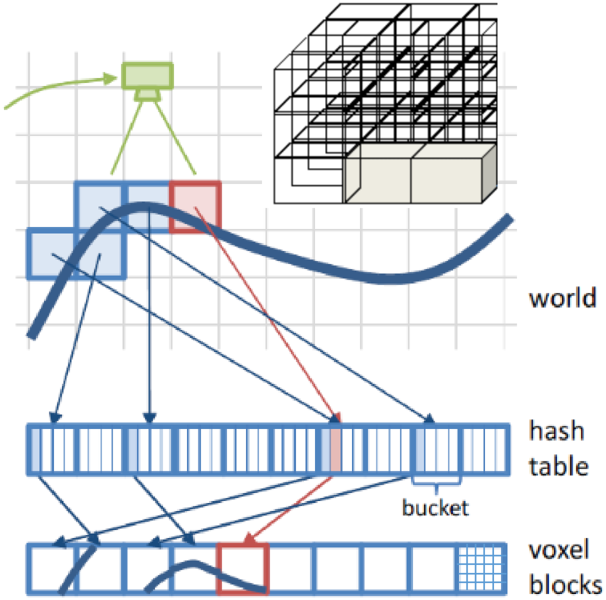


- Sparse
- Structural
- Indirect Index Query

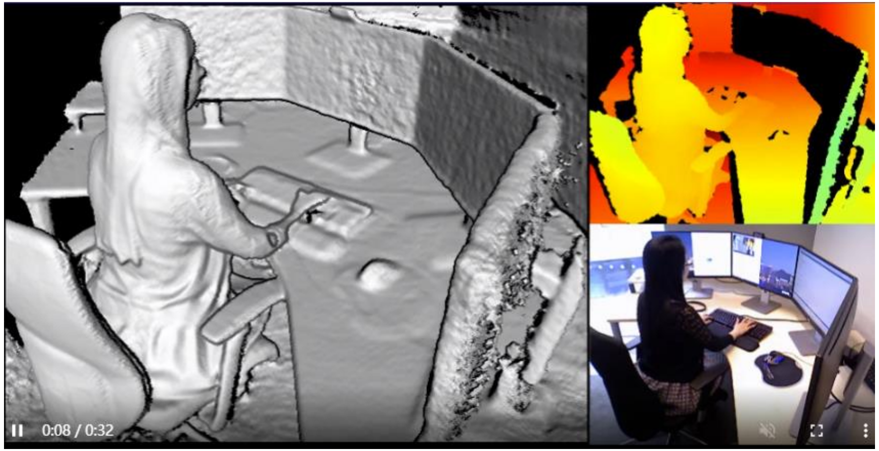


https://octomap.github.io/

Voxel hashing



- Most Sparse
- Structural
- Indirect Index Query



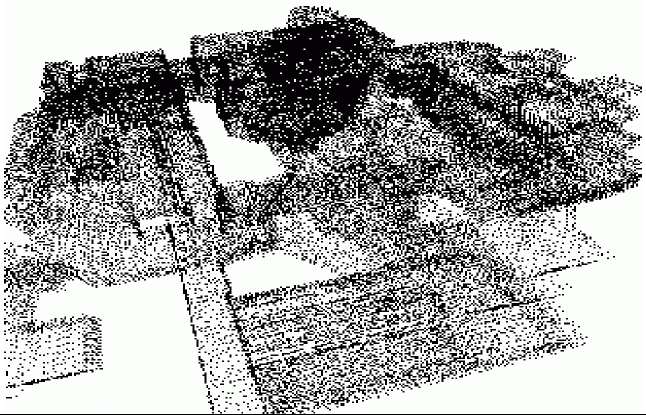
Voxel Hashing:

https://github.com/niessner/VoxelHashing

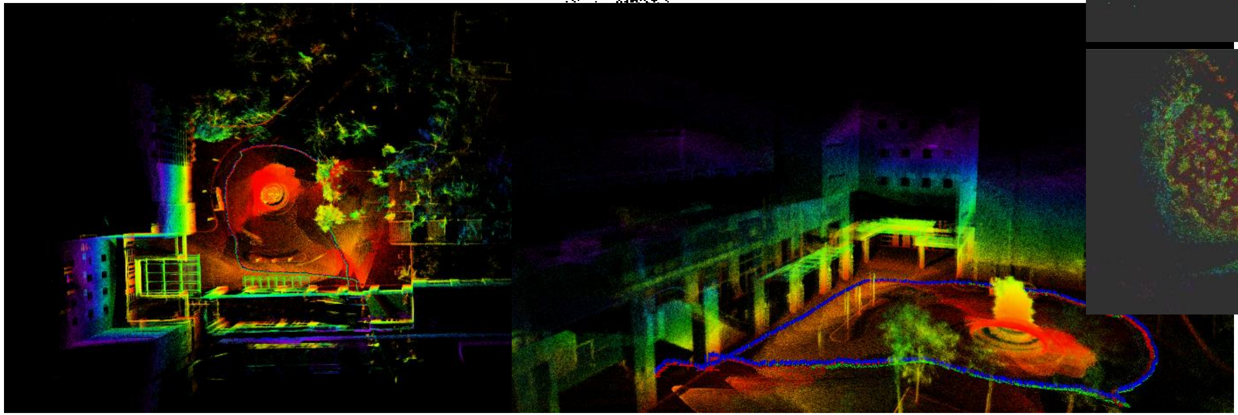
InfiniTAM:

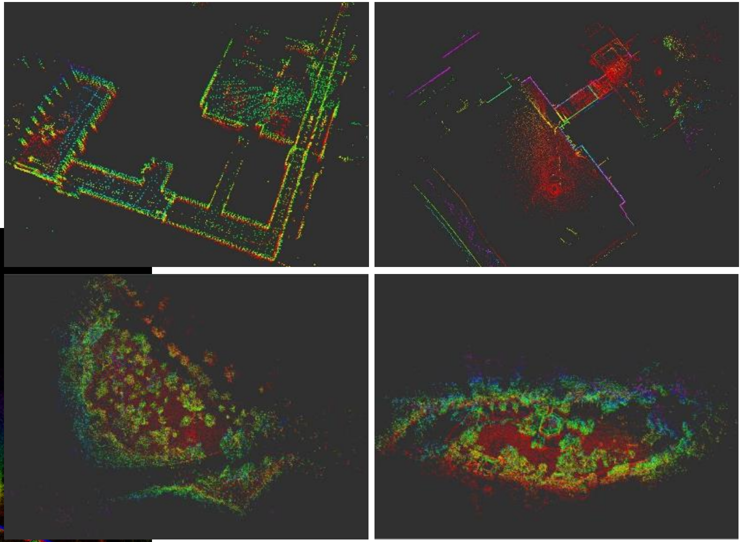
http://www.robots.ox.ac.uk/~victor/infinitam/

Point cloud map



- Un-ordered
- No Index Query



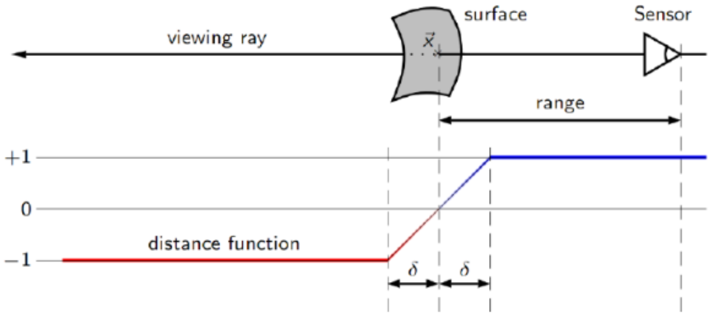


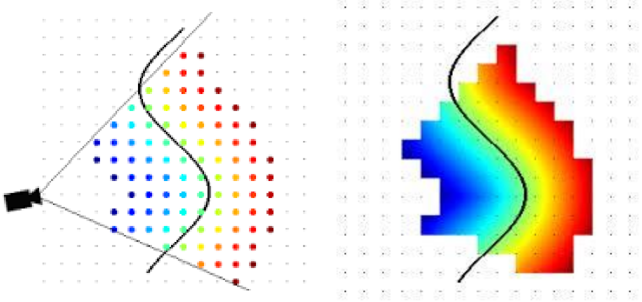
PCL

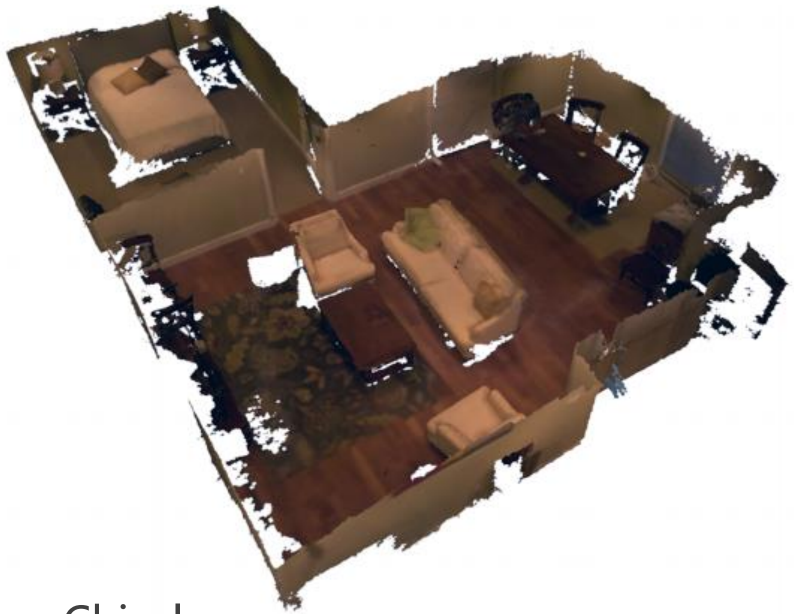
http://pointclouds.org/

TSDF map

Truncated Signed Distance Functions







OpenChisel

https://github.com/personalrobotics/OpenChisel

ESDF map

Euclidean Signed Distance Functions

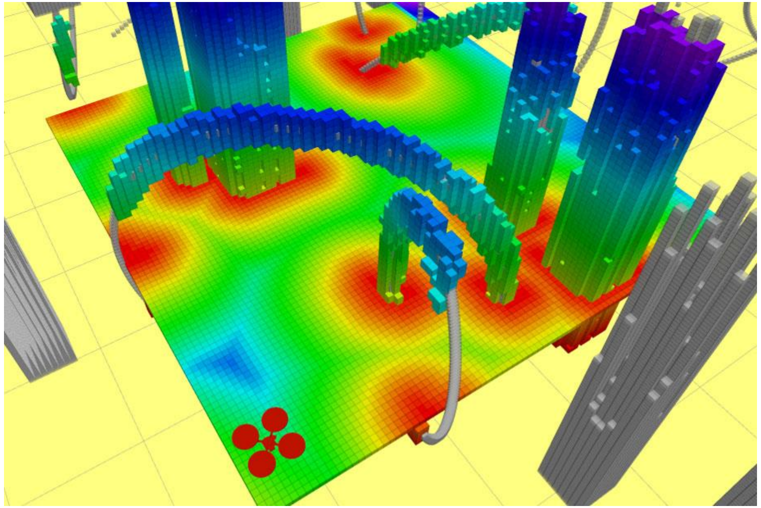
Incremental Update, Global Map

Batch Update, Local Map

Running the Cow_and_Lady Dataset[1]
Compare with Voxblox[2]

[1] https://projects.asl.ethz.ch/datasets/doku.php?id=iros2017/

[2] Helen Oleynikova, Zachary Taylor, Marius Fehr, Juan Nieto, and Roland Siegwart, "Voxblox: Building 3D Signed Distance Fields for Planning", In IEEE Int. Conf. on Intelligent Robots and Systems (IROS), October 2017.



Distance Transforms of Sampled Functions, PF Felzenszwalb

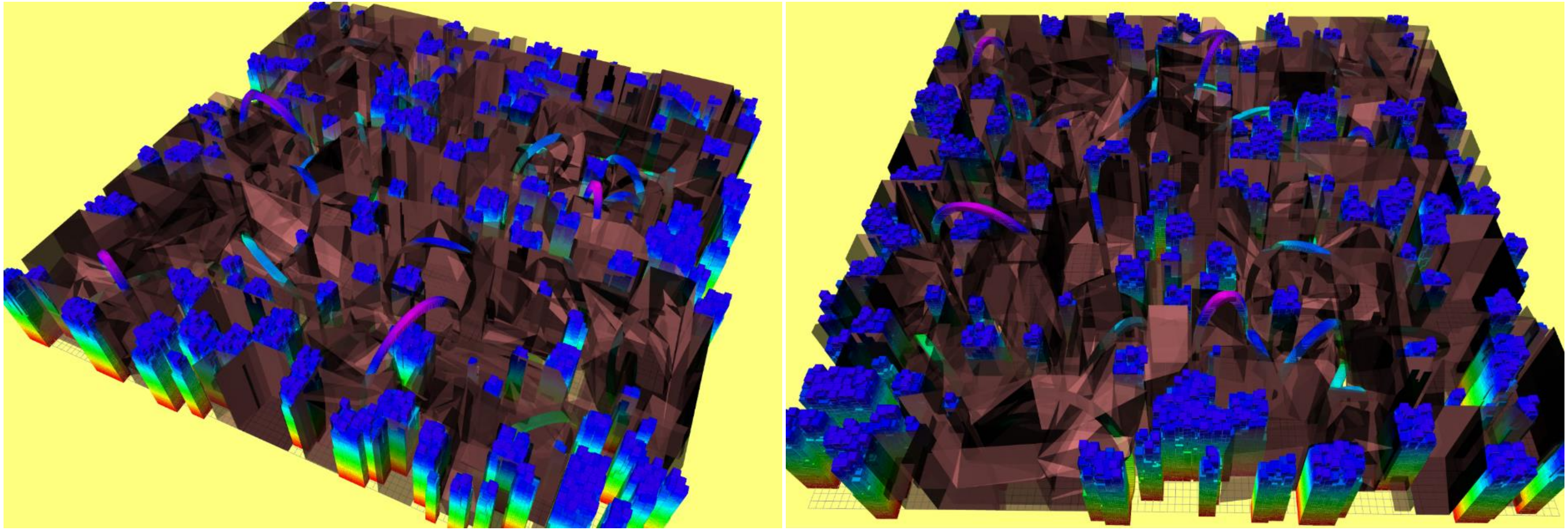
VoxBlox
https://github.com/ethz-asl/voxblox

FIESTA
https://github.com/HKUST-Aerial-Robotics/FIESTA

TRR's Local Map
https://github.com/HKUST-Aerial-Robotics/Teach-Repeat-Replan

More ?

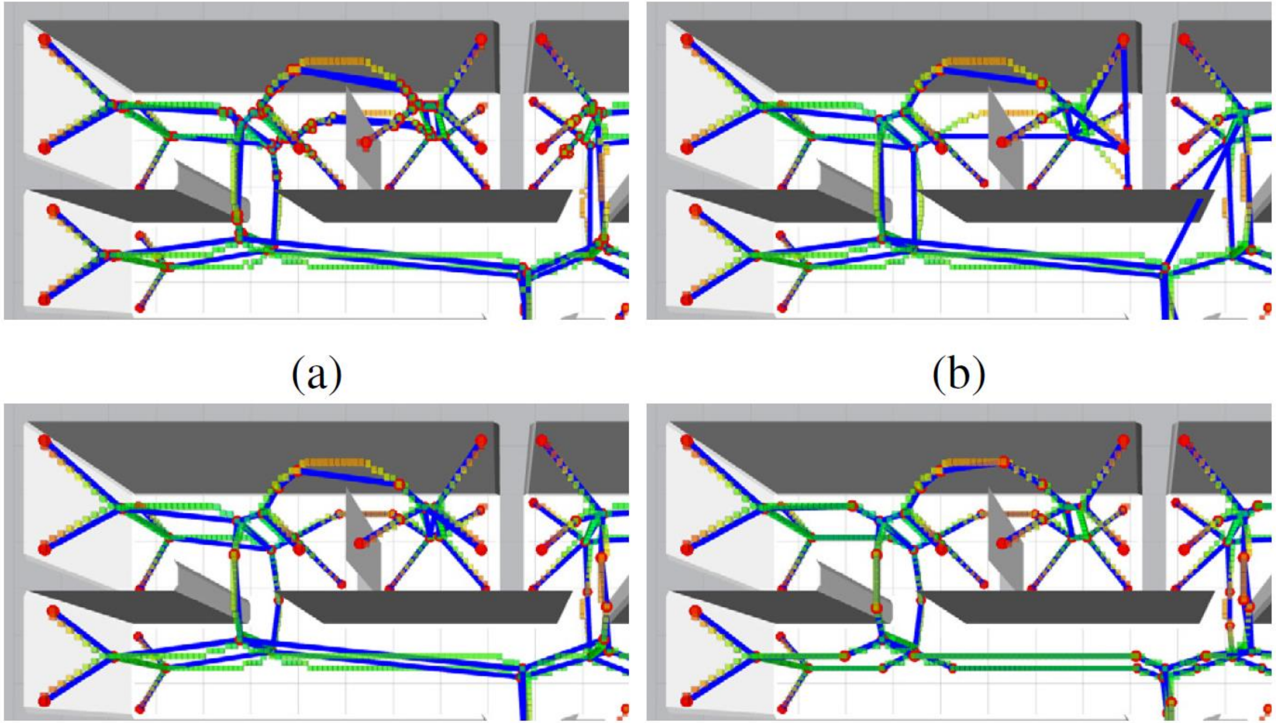
Free-space Roadmap



https://github.com/HKUST-Aerial-Robotics/Teach-Repeat-Replan

More ?

Voronoi Diagram Map



https://github.com/ethz-asl/mav_voxblox_planning

Pre-requirement

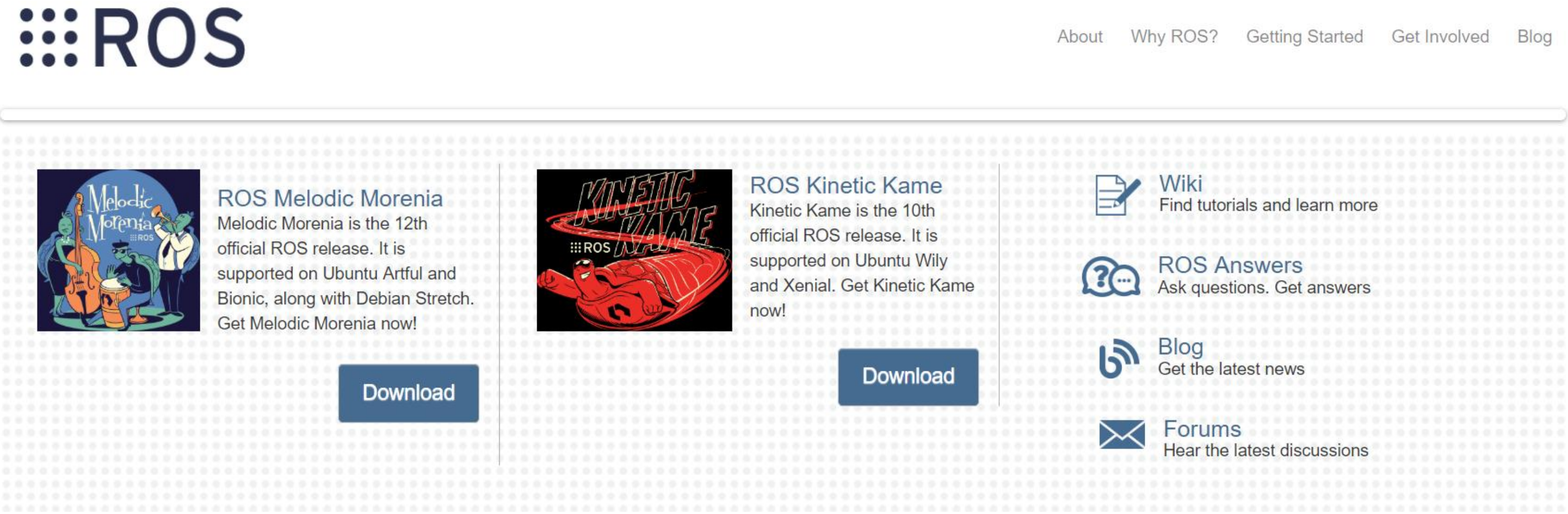
Linux

- Linux file system
- How to install software in linux
- Useful commands

C++ and GCC Toolchain

- C with class ?
- Gcc, Makefile, CMakeList
- Write CMakeList
- How to solve problems: google and document

ROS



- Follow ROS tutorial
- Ubuntu 16.04 + ros kinetic is recommended

Matlab

- Please install Matlab

Homework

Thanks for Listening!