ELEC 3210
Introduction to Mobile Robotics
Lecture 18

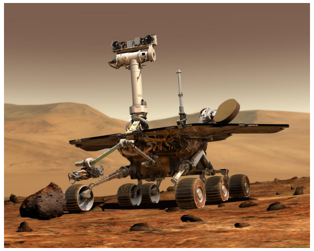
(Machine Learning and Information Processing for Robotics)

Speaker: Haokun WANG

E-mail: hwangeh@connect.ust.hk

Ph.D. Candidate, Dept. of IIP (ROAS)

Supervised by Prof. Shaojie Shen



I am interested in **robotics motion planning**, particularly in finding elegant solutions for contact-based robotic interaction.

EDUCATION

Sep 2020 - Present — **Hong Kong University of Science and Technology** — **Hong Kong**
Ph.D. candidate at HKUST Robotics Institute

Sep 2015 – Jun 2019 — **Southern University of Science and Technology** — **Shenzhen**
Bachelor of Computer Science and Technology

RESEARCH EXPERIENCE

Mar. 2022 – Mar. 2023 — **Visiting Student,** *FAST Lab*, ZJU — **Huzhou**

- Aerial robots with suspended payload: Designed motion planning algorithms and control methods for autonomous generation of slack-taut mode switching in the cable-payload system. Validated the research through simulations and physical experiments. Research outcomes submitted to top journals in the field.

Jul. 2018 – Sep. 2018 — **Visiting Student,** *Collaborative Advanced Robotics and Intelligent System Laboratory (CARIS)*, UBC — **Vancouver**

- DLR Gripper Project: By leveraging the Universal Robotic Description Format (URDF) and Robotic Operation System (ROS), the project aims to create an accurate representation of the KUKA arm and DLR gripper, enabling simulations and experiments in Gazebo environment.

Jul. 2016 – Jun. 2019 — **Undergraduate Student,** *Intelligent Sensing and Unmanned System Laboratory (ISUS)*, SUSTech — **Shenzhen**

- Multi-UAV collaboration system in simulation (innovation training): The system enables multiple UAVs to work together effectively, employing vision algorithms for assisted positioning. By utilizing the V-REP simulation, the project refine the collaborative capabilities of the UAVs, fostering innovation in aerial robots.
- Semantic segmentation using weakly- and semi-supervised learning in complex scenes (graduate project): Developed an iterative semantic segmentation framework based on GrabCut and DeepLab methods for complex scene analysis. Implemented human segmentation in challenging environments using weakly- and semi-supervised learning techniques.

PROFESSIONAL EXPERIENCE

Jul. 2019 – Jul. 2020 — **Research Assistant,** SUSTech Institution of Robotics (SIR) — **Shenzhen**

- DeepClaw benchmarking for manipulation: Developed DeepClaw, an application of deep learning in robot manipulation research. Designed a Python-based software platform that supports easy expansion to multiple hardware setups. Implemented various manipulation and interaction tasks, including playing Tic-Tac-Toe, solving jigsaw puzzles, and more.
- Reconfigurable design for omni-adaptive grasp learning: Implemented the entire experimental process, including data collection and vision detection, for various configurations of soft fingers with different numbers and directions. Validated these designs using the YCB Objects dataset.

Sep. 2018 – Dec. 2018 — **Computer Vision Research Intern,** Malong Technologies — **Shenzhen**

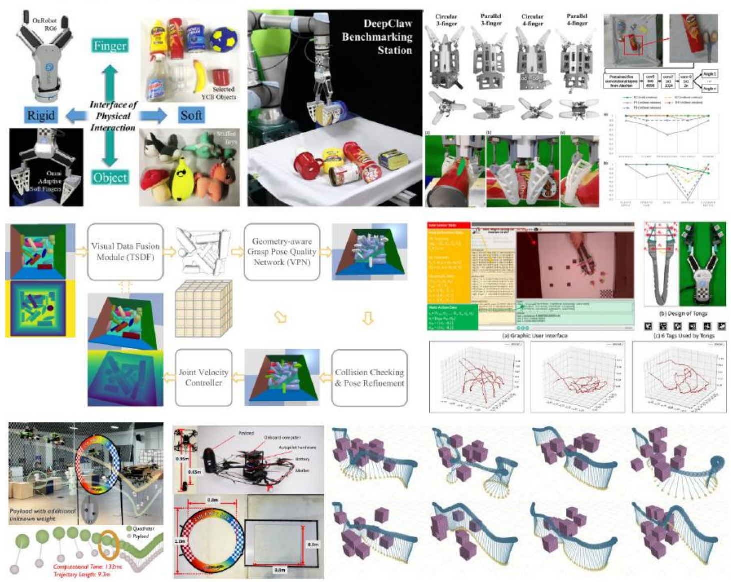
- Deep learning-based fine-grained product detection and recognition: Integrated deep learning-based visual detection algorithms into unmanned vending machines. Trained and tested multi-class segmentation and detection algorithms on a custom dataset to achieve fine-grained product recognition. The outcomes were incorporated as part of a bachelor's thesis.

Jul. 2017 – Sep. 2017 — **Robotics Research Intern,** Xeno Dynamics — **Shenzhen**

- Self-balancing service robot project: Investigated the dynamics theory of a two-wheeled self-balancing robot. Designed and implemented a closed-loop multi-level PID control algorithm based on data from IMU and magnetic encoders.

PUBLLICATIONS

- **H. Wang**, H. Li, B. Zhou, F. Gao, and S. Shen, "Impact-Aware Motion Planning and Control for Aerial Robots with Suspended Payloads," submitted to IEEE Transaction on Robotics (T-RO).
- H. Li, **H. Wang**, C. Feng, F. Gao, B. Zhou and S. Shen, "AutoTrans: A Complete Planning and Control Framework for Autonomous UAV Payload Transportation," in **IEEE Robotics and Automation Letters**, vol. 8, no. 10, pp. 6859-6866, Oct. 2023, doi: 10.1109/LRA.2023.3313010.
- **H. Wang**, X. Liu, N. Qiu, N. Guo, F. Wan, and C. Song, "Deepclaw 2.0: A data collection platform for learning human manipulation," **Frontiers in Robotics and AI**, vol. 9, 2022.
- J. Cai, J. Cen, **H. Wang** and M. Y. Wang, "Real-Time Collision-Free Grasp Pose Detection with Geometry-Aware Refinement Using High-Resolution Volume," in **IEEE Robotics and Automation Letters**, vol. 7, no. 2, pp. 1888-1895, April 2022, doi: 10.1109/LRA.2022.3142424.
- F. Wan, **H. Wang**, X. Liu, L. Yang and C. Song, "DeepClaw: A Robotic Hardware Benchmarking Platform for Learning Object Manipulation," 2020 **IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)**, Boston, MA, USA, 2020, pp. 2011-2018, doi: 10.1109/AIM43001.2020.9159011.
- F. Wan, **H. Wang**, J. Wu, Y. Liu, S. Ge and C. Song, "A Reconfigurable Design for Omni-Adaptive Grasp Learning," in **IEEE Robotics and Automation Letters**, vol. 5, no. 3, pp. 4210-4217, July 2020, doi: 10.1109/LRA.2020.2982059.
- L. Yang, F. Wan, **H. Wang**, X. Liu. Y. Liu, J. Pan, and C. Song, "Rigid-Soft Interactive Learning for Robust Grasping," in **IEEE Robotics and Automation Letters**, vol. 5, no. 2, pp. 1720-1727, April 2020, doi: 10.1109/LRA.2020.2969932.



Contents

Introduction

- Background & Motivation
- Preliminary

Background & Motivation

Paul Gauguin

"Where Do We Come From? What Are We? <u>Where Are We Going</u>?"

Background & Motivation

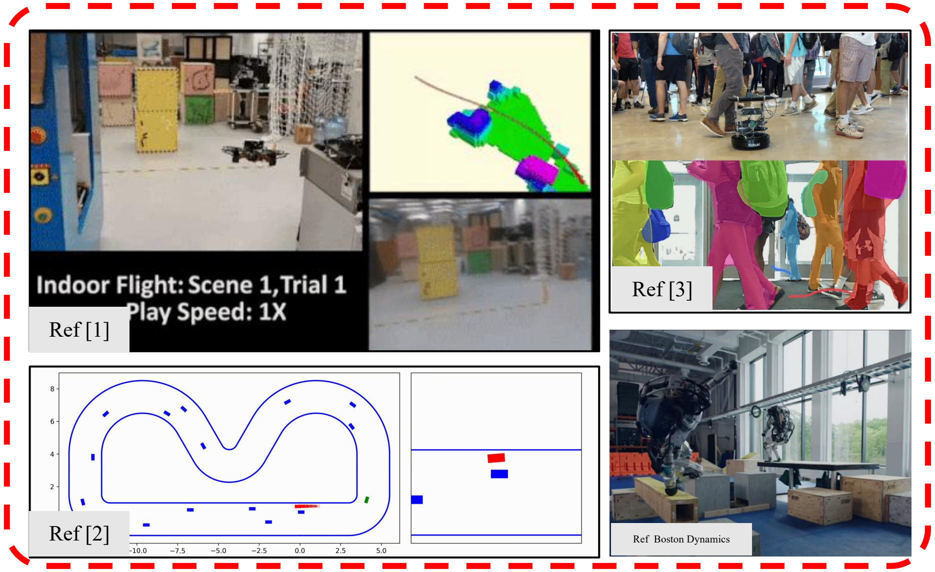
Paul Gauguin

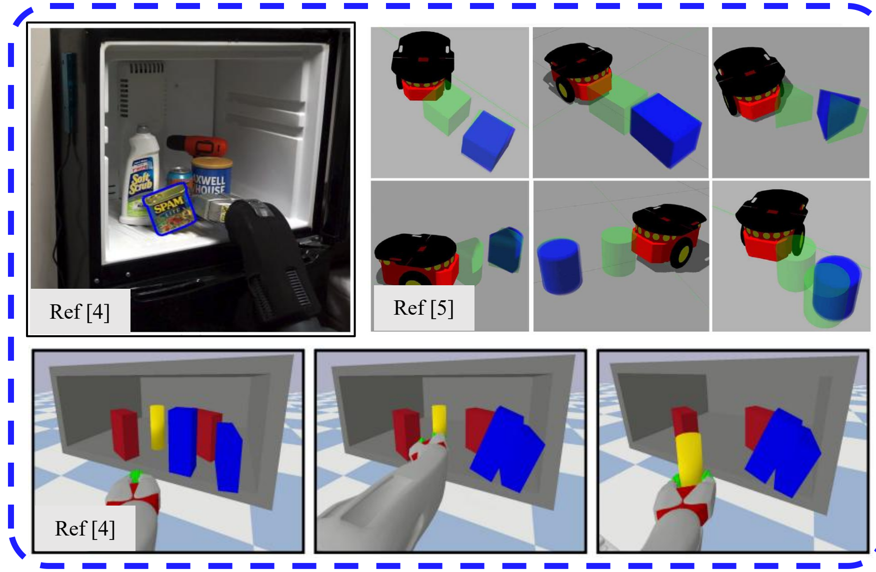
"Where Do We Come From? What Are We? <u>Where Are We Going</u>?"

Raw Data → Perception / Estimation → Decision / Planning → Execution / Control → Actions

Background & Motivation

Motion Planning of Mobile Robots:





Interaction:
- Obstacle avoidance
- Contact-aware
- …

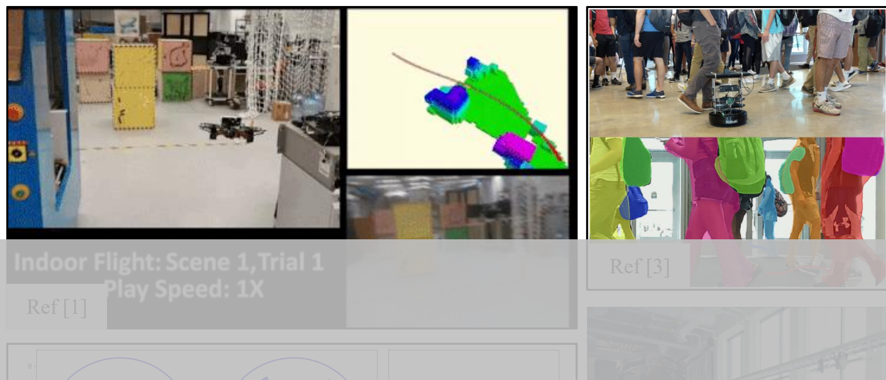
System dynamics:
- Physical limitations
- Feasibility
- Safety
- …

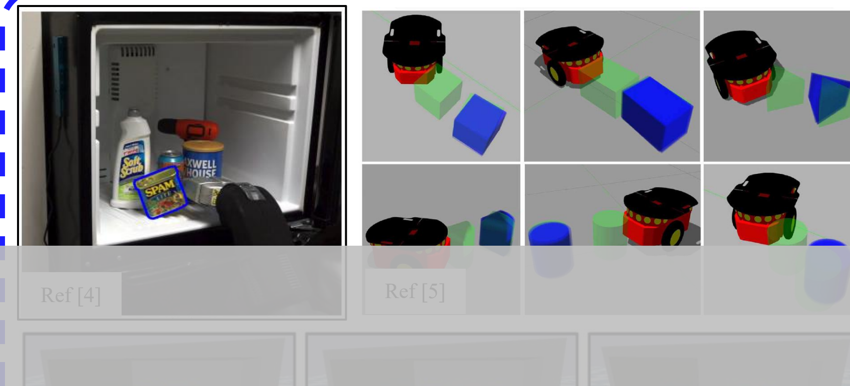
Mission goals:
- Navigation
- Pick-and-place
- Non-prehensile manipulation
- …

[1] B. Zhou, F. Gao, L. Wang, C. Liu and S. Shen, "Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight," in IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 3529-3536, Oct. 2019, doi: 10.1109/LRA.2019.2927938.
[2] S. He, J. Zeng and K. Sreenath, "Autonomous Racing with Multiple Vehicles using a Parallelized Optimization with Safety Guarantee using Control Barrier Functions," 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 3444-3451, doi: 10.1109/ICRA46639.2022.9811969.
[3] A. J. Sathyamoorthy, J. Liang, U. Patel, T. Guan, R. Chandra and D. Manocha, "DenseCAvoid: Real-time Navigation in Dense Crowds using Anticipatory Behaviors," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 11345-11352, doi: 10.1109/ICRA40945.2020.9197379.
[4] D. M. Saxena, and M. Likhachev, "Planning for Complex Non-prehensile Manipulation Among Movable Objects by Interleaving Multi-Agent Pathfinding and Physics-Based Simulation, " 2023 arXiv preprint arXiv:2303.13352.
[5] J. St¨uber, C. Zito, and R. Stolkin, “Let’s push things forward: A survey on robot pushing,” Frontiers in Robotics and AI, vol. 7, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2020.00008

Background & Motivation

Motion Planning of Mobile Robots:





Fundamental requirement: to generate *collision-free* and *smooth* trajectories *from start to goal* for autonomous mobile robots.

- Contact-aware
- …

- Feasibility
- Safety
- …

- Pick-and-place
- Non-prehensile manipulation
- …

[1] B. Zhou, F. Gao, L. Wang, C. Liu and S. Shen, "Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight," in IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 3529-3536, Oct. 2019, doi: 10.1109/LRA.2019.2927938.
[2] S. He, J. Zeng and K. Sreenath, "Autonomous Racing with Multiple Vehicles using a Parallelized Optimization with Safety Guarantee using Control Barrier Functions," 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 3444-3451, doi: 10.1109/ICRA46639.2022.9811969.
[3] A. J. Sathyamoorthy, J. Liang, U. Patel, T. Guan, R. Chandra and D. Manocha, "DenseCAvoid: Real-time Navigation in Dense Crowds using Anticipatory Behaviors," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 11345-11352, doi: 10.1109/ICRA40945.2020.9197379.
[4] D. M. Saxena, and M. Likhachev, "Planning for Complex Non-prehensile Manipulation Among Movable Objects by Interleaving Multi-Agent Pathfinding and Physics-Based Simulation, " 2023 arXiv preprint arXiv:2303.13352.
[5] J. St¨uber, C. Zito, and R. Stolkin, "Let's push things forward: A survey on robot pushing," Frontiers in Robotics and AI, vol. 7, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2020.00008

Preliminary: Environment Representation

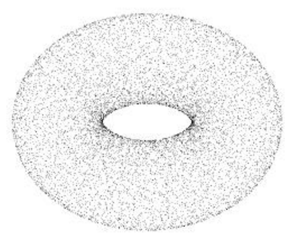
Raw Data

Interoceptive Sensors:
- Rotary encoders
- Accelerometers
- Inertial measurement devices
- …

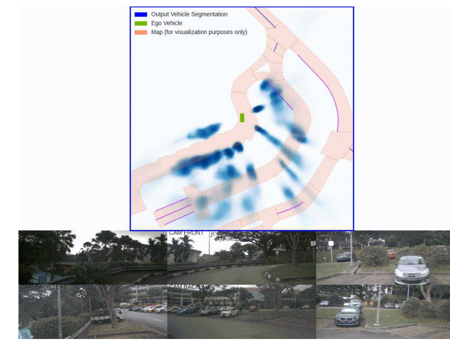
Exteroceptive Sensors:
- LiDAR
- Sonar
- Radar
- Cameras
- …

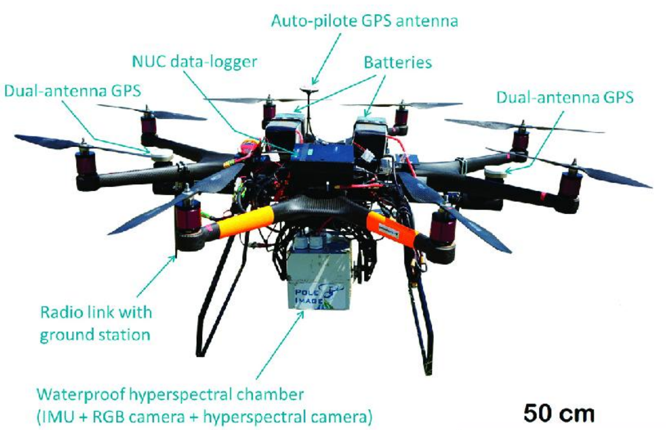
Data Representation

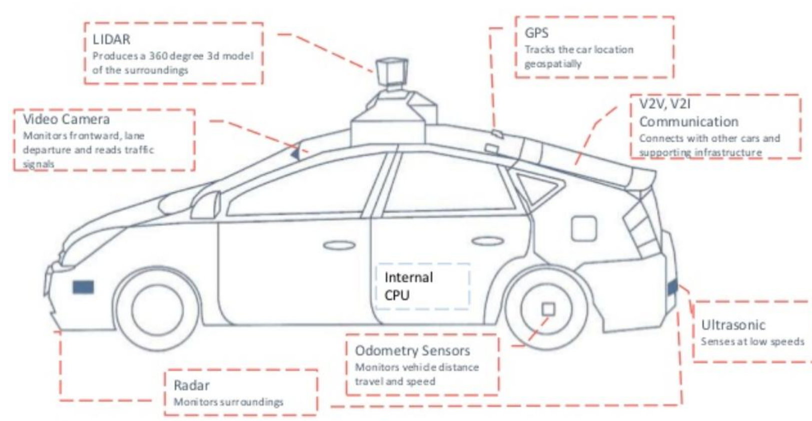
Unstructured Representation V.S. Structured Representation


Point cloud[1]


Occupancy grid mapping[2]


BEV mapping[3]

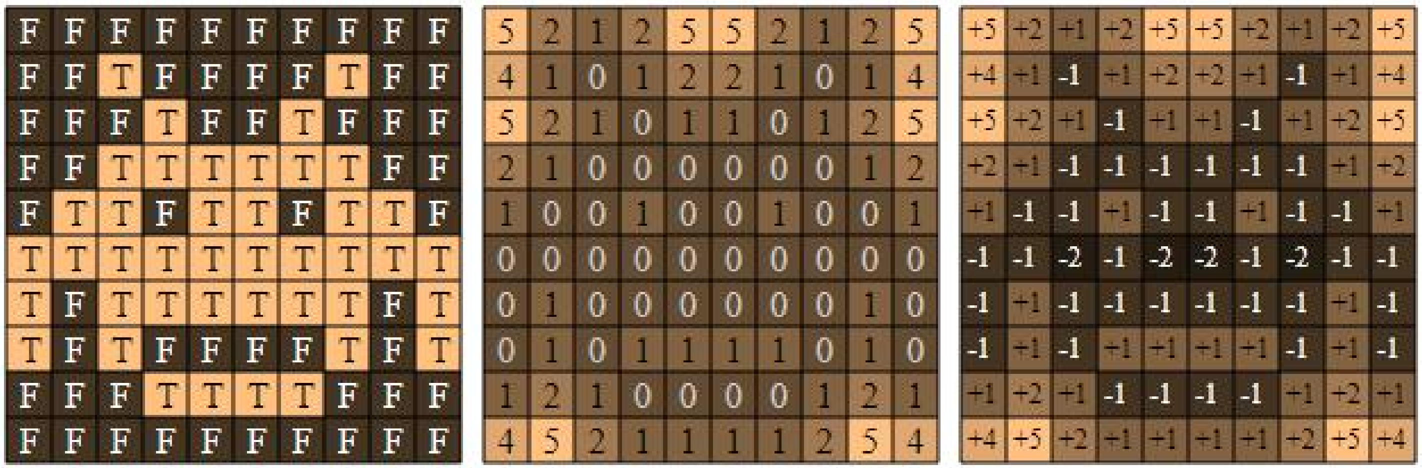




[1] https://en.wikipedia.org/wiki/Point_cloud
[2] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “OctoMap: An efficient probabilistic 3D mapping framework based on octrees,” Autonomous Robots, 2013, software available at https://octomap.github.io. [Online]. Available: https://octomap.github.io.
[3] https://github.com/RuslanAgishev/bev-net

Preliminary: Environment Representation

Euclidean Distance Transform



(a) (b) (c)

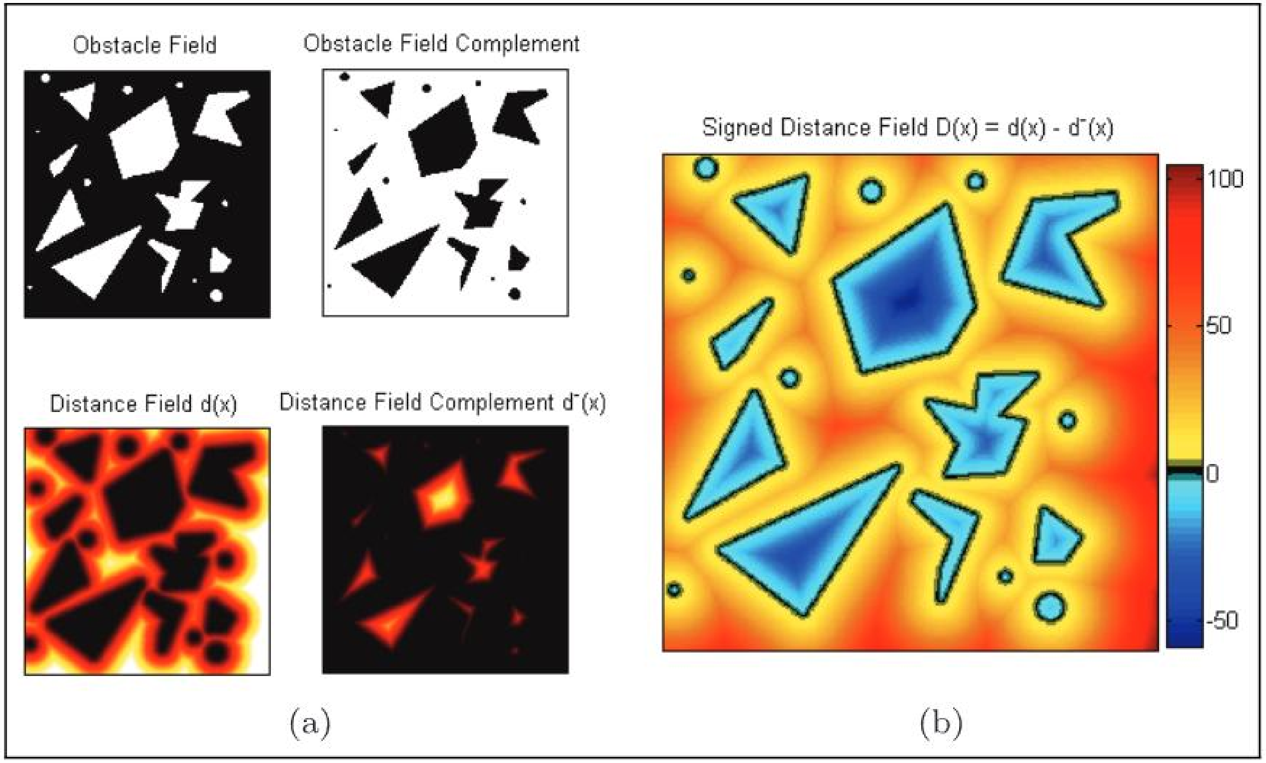
(a) Input Boolean field, (b) squared Euclidean distance, and (c) signed distance field.

- The **EDT** (Euclidean Distance Transform) can be defined as consuming a field of booleans and producing a field of scalars such that each value in the output is the distance to the nearest "true" cell in the input.
- A useful concept is the **SDF** (signed distance field). Note that negative values are inside the contour of the shape and positive values are outside.

[1] Reference: https://prideout.net/blog/distance_fields/

Preliminary: Environment Representation

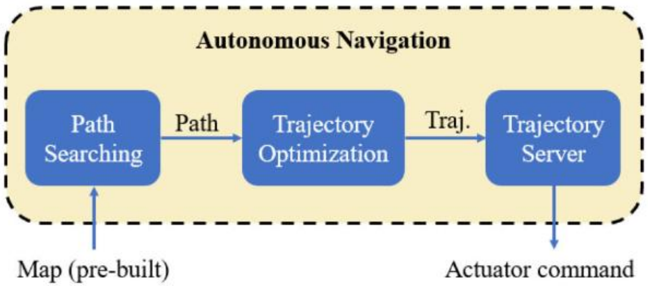
SDF Map



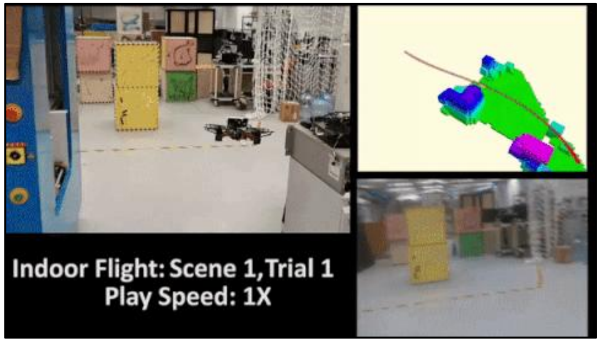
- The signed distance field is created as the difference between two distance transforms created over a binary grid where obstacles are set to 1, and its complement, where obstacles are set to zero, so that $\mathcal{D}(x) = d(x) - \overline{d}(x)$. Here, brighter colors represent higher values.

[1] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," The International journal of robotics research, vol. 32, no. 9-10, pp. 1164–1193, 2013.

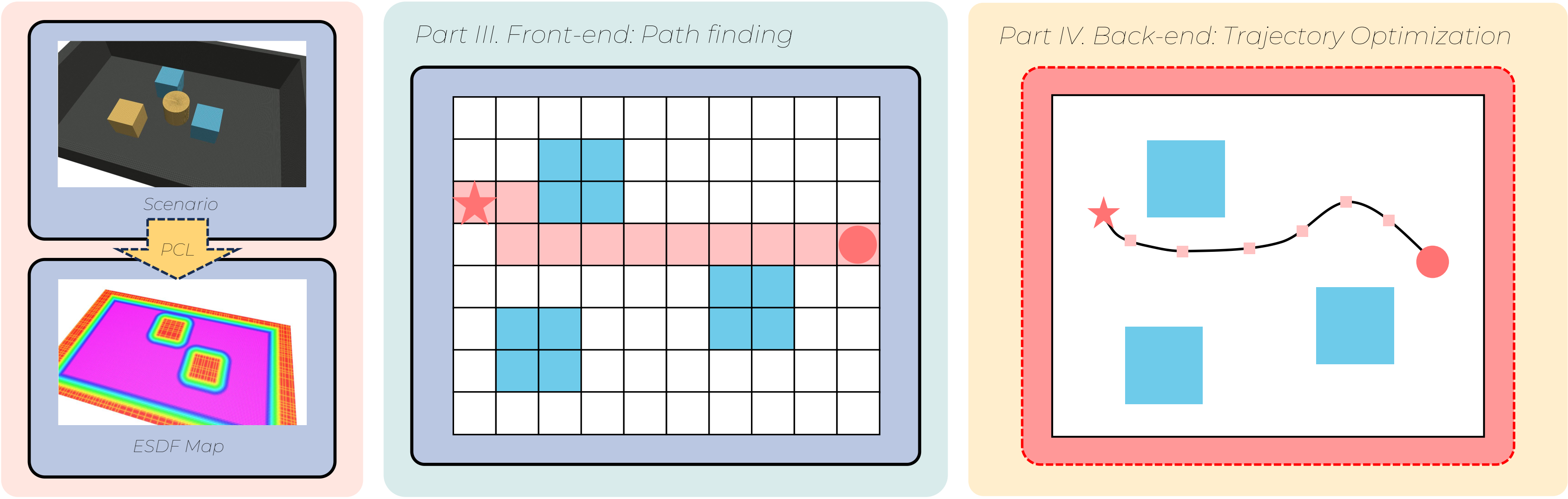
Preliminary: Motion Planning Framework



[1]

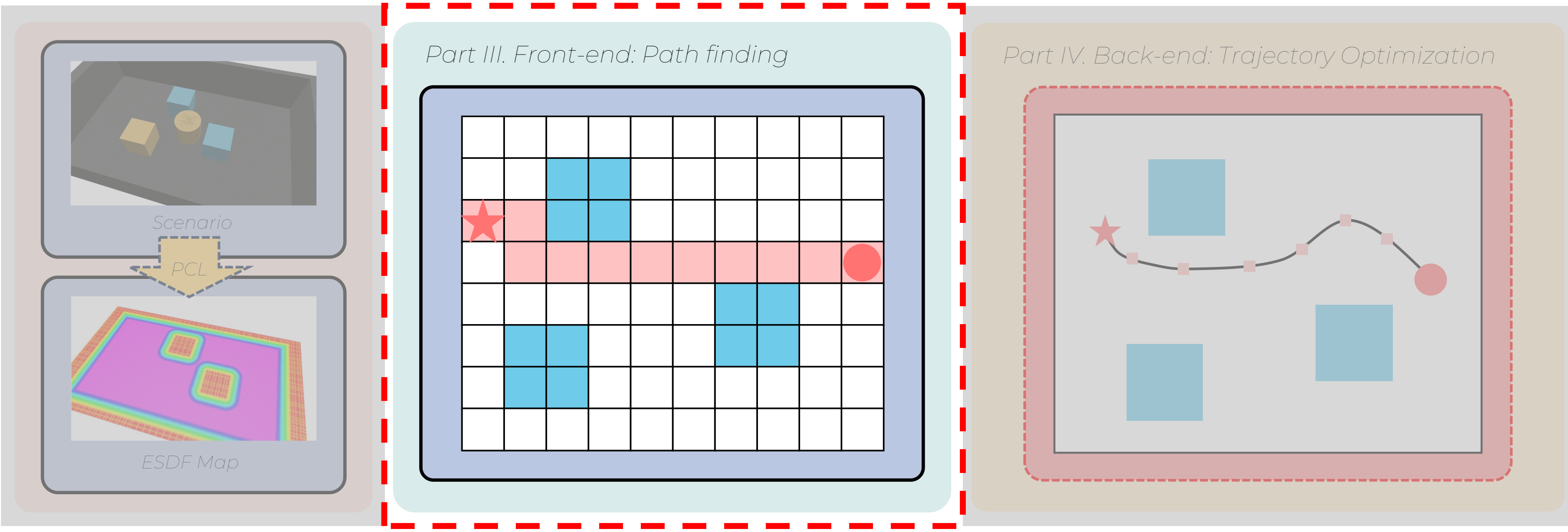


[2]



[1] L. Quan, L. Han, B. Zhou, S. Shen, and F. Gao, "Survey of uav motion planning," IET Cyber-Systems and Robotics, vol. 2, no. 1, pp. 14–21, 2020. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-csr.2020.0004

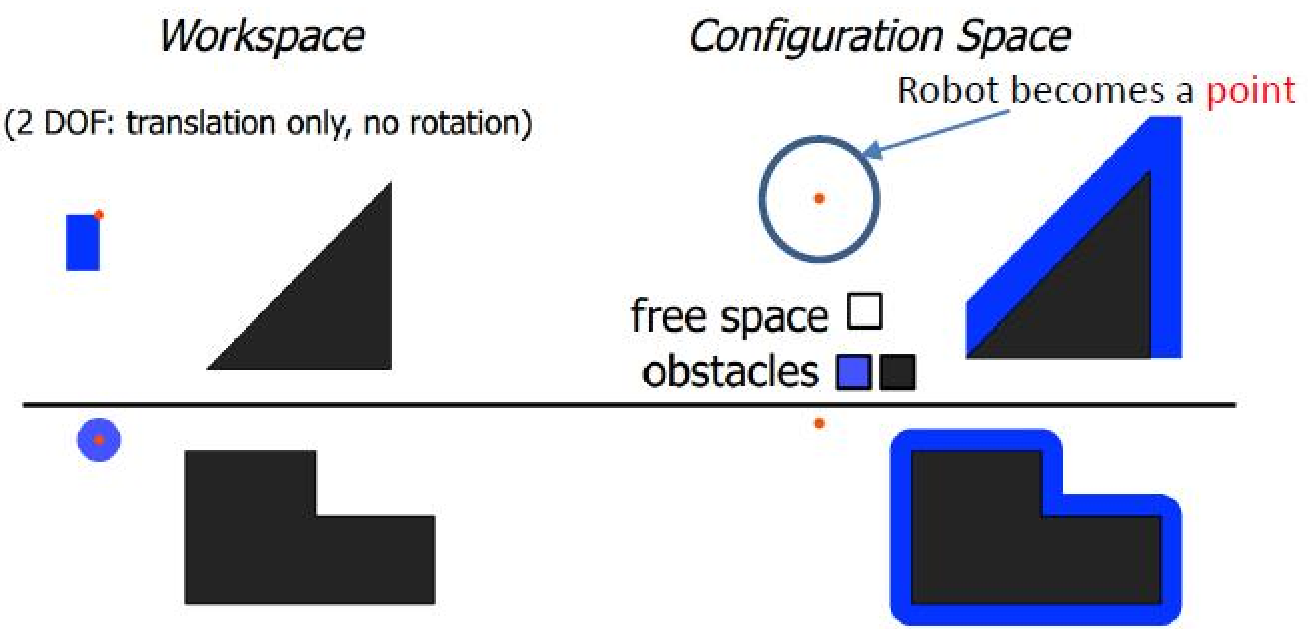
[2] B. Zhou, F. Gao, L. Wang, C. Liu and S. Shen, "Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight," in IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 3529-3536, Oct. 2019, doi: 10.1109/LRA.2019.2927938.



Path Finding Methods

- Configuration Space
- Sampling-Based Methods
- Searching-Based Methods

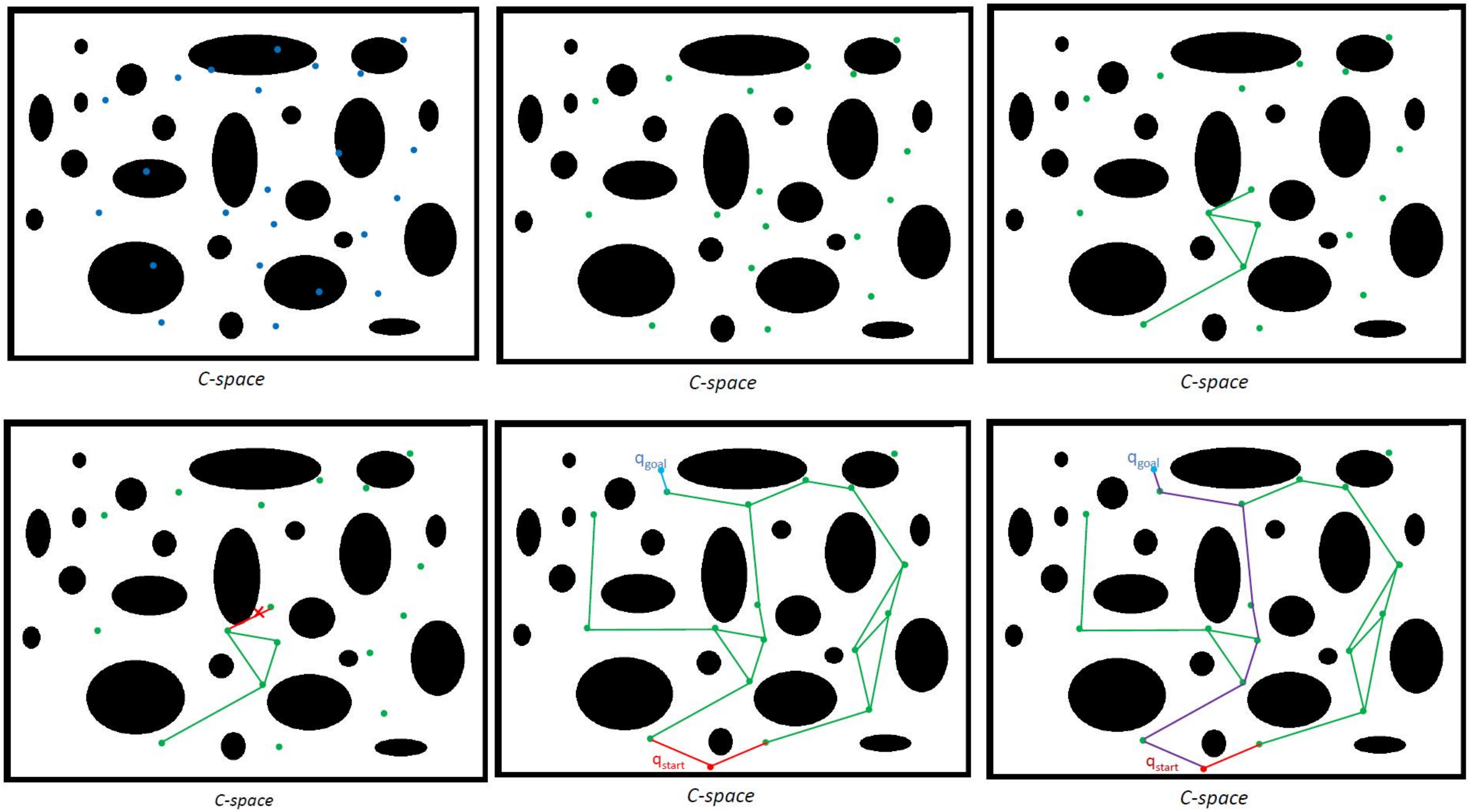
Configuration Space



- The **configuration** of a robot is a complete specification of the position of every point of the robot.
- The *n*-dimensional space containing all possible configurations of the robot is called the **configuration space (C-space**). The configuration of a robot is represented by a point in its C-space.
- Representing an obstacle in C-space can <u>be extremely complicated</u>. So approximated (but more conservative) representations are used in practice.

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

Probabilistic Roadmap (PRM)



[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

Probabilistic Roadmap (PRM)

```
(1)   N ← ∅
(2)   E ← ∅
(3)   loop
(4)       c ← a randomly chosen free
           configuration
(5)       N_c ← a set of candidate neighbors
           of c chosen from N
(6)       N ← N ∪ {c}
(7)       for all n ∈ N_c, in order of
           increasing D(c,n) do
(8)         if ¬same_connected_component(c,n)
             ∧Δ(c,n) then
(9)                 E ← E ∪ {(c,n)}
(10)                update R's connected
                     components
```

✓ Probabilistically complete: i.e., with probability one, if fun for long enough the graph will contain a solution path if one exists.

✓ Can cope with high-dimensional system.

- Collision detection takes majority of time.
- Suboptimal solution if only limited samples are given.
- Build graph over C-space but no particular focus on generating a path.

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

Rapidly Exploring Random Trees

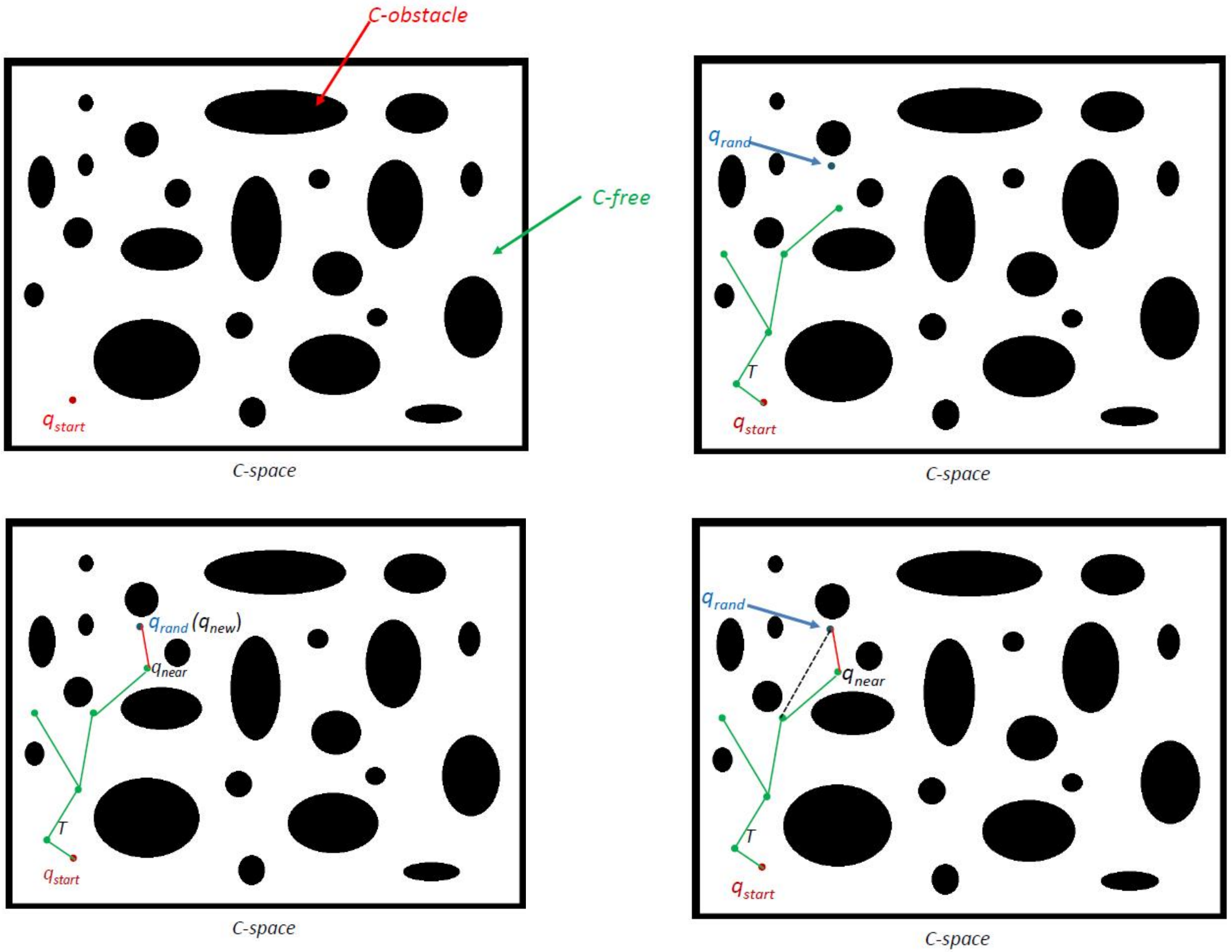
Algorithm BuildRRT
Input: Start configuration q_{start}, number of vertices in RRT K
Output: RRT T
L1: G.init(q_{start})
L2: **for** k = 1 **to** K
L3: $q_{rand} \leftarrow$ RAND_CONF();
L4: $q_{near} \leftarrow$ NEAREST_VERTEX(q_{rand}, T);
L5: $q_{new} \leftarrow$ NEW_CONF(q_{near}, q_{rand});
L6: T.add_vertex(q_{new}); T.add_edge(q_{near}, q_{new})
L7: **return** G

Basic Idea:

- Starting from the start configuration, build up a tree through generating "next configuration".

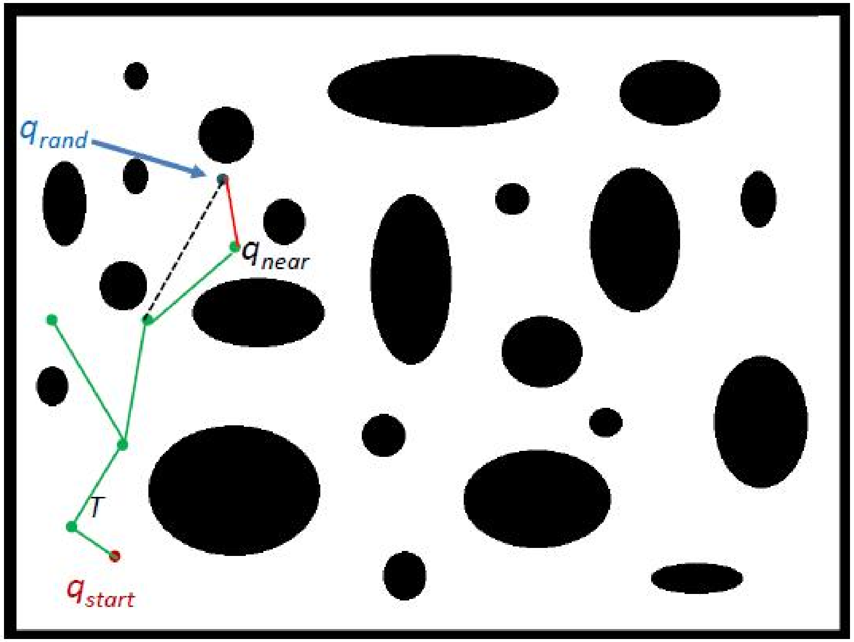
[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

Rapidly Exploring Random Trees



[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

RRT*



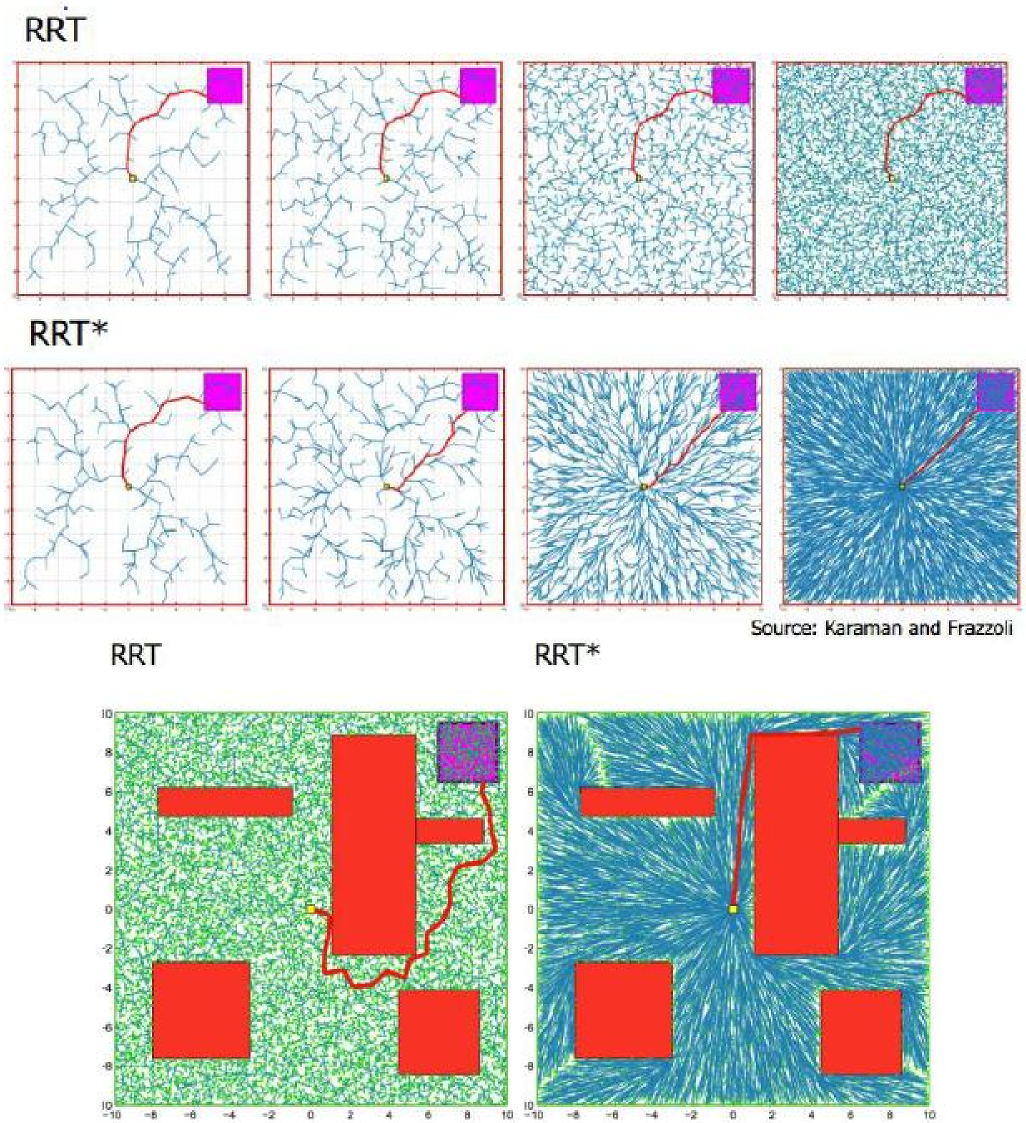
C-space

Basic Idea:

- RRT is simple but is prone to be probabilistic incomplete.
- Add **rewire** function: swap new point in as parent for nearby vertices who can be reached along shorter path through new point than through their original path.
- RRT* is asymptotically optimal.

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

RRT*

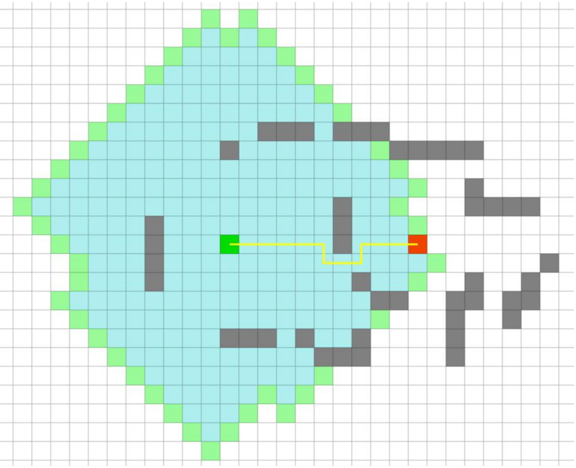


[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

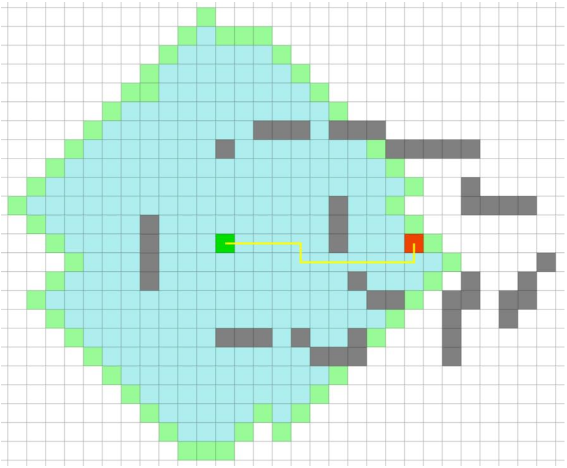
Searching-Based Methods

Basic Idea:

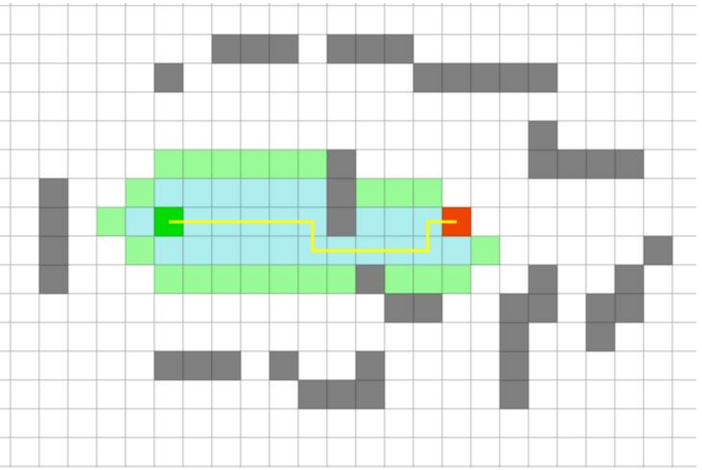
- ➢ Construct a state space graph, which is a mathematical representation of a **search algorithm**.
- ➢ For every search problem, there's a corresponding state space graph.
- ➢ Connectivity between nodes in the graph is represented by edges.



Breadth-First Searching



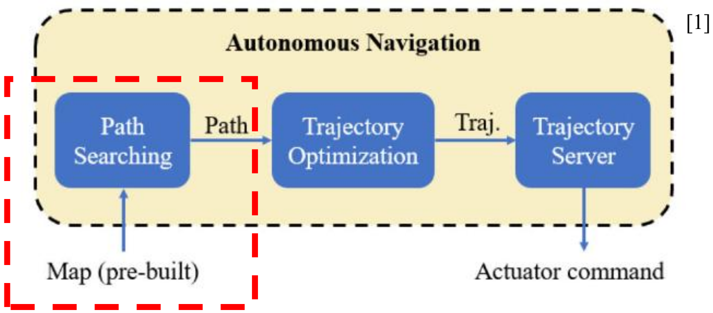
Dijkstra Method

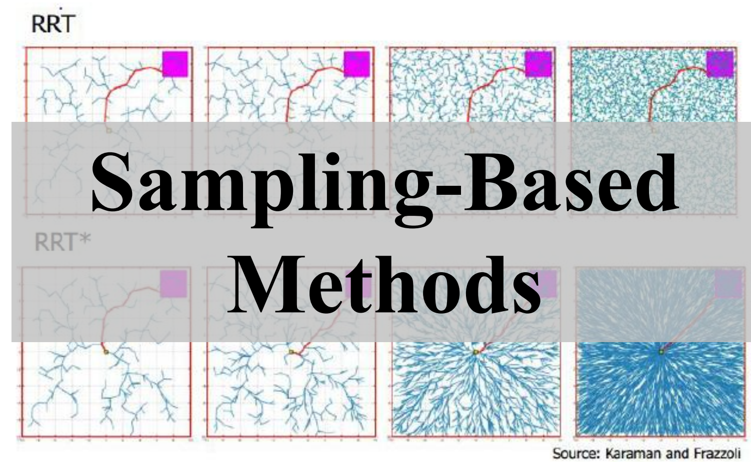


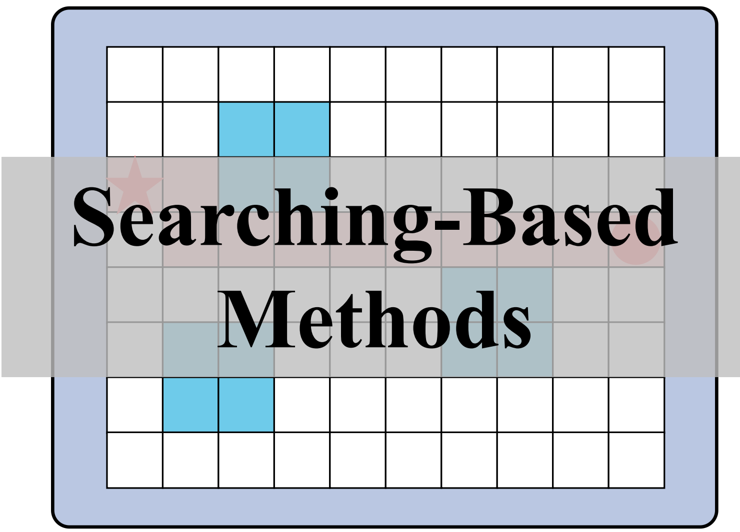
A* Searching

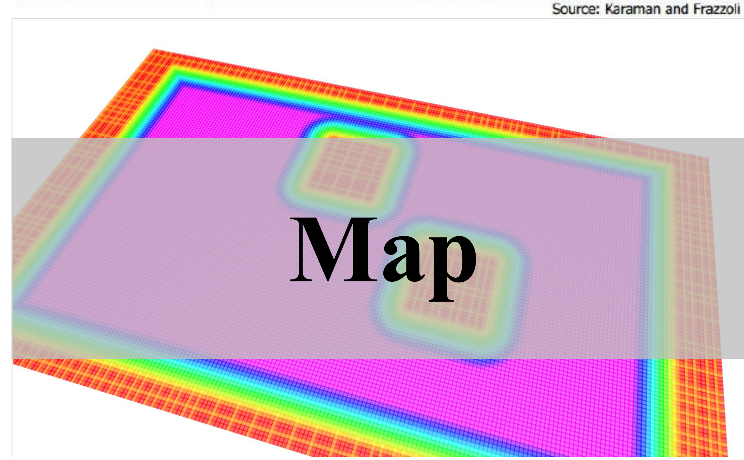
[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf
[2] Reference: https://qiao.github.io/PathFinding.js/visual/

Path Finding Methods





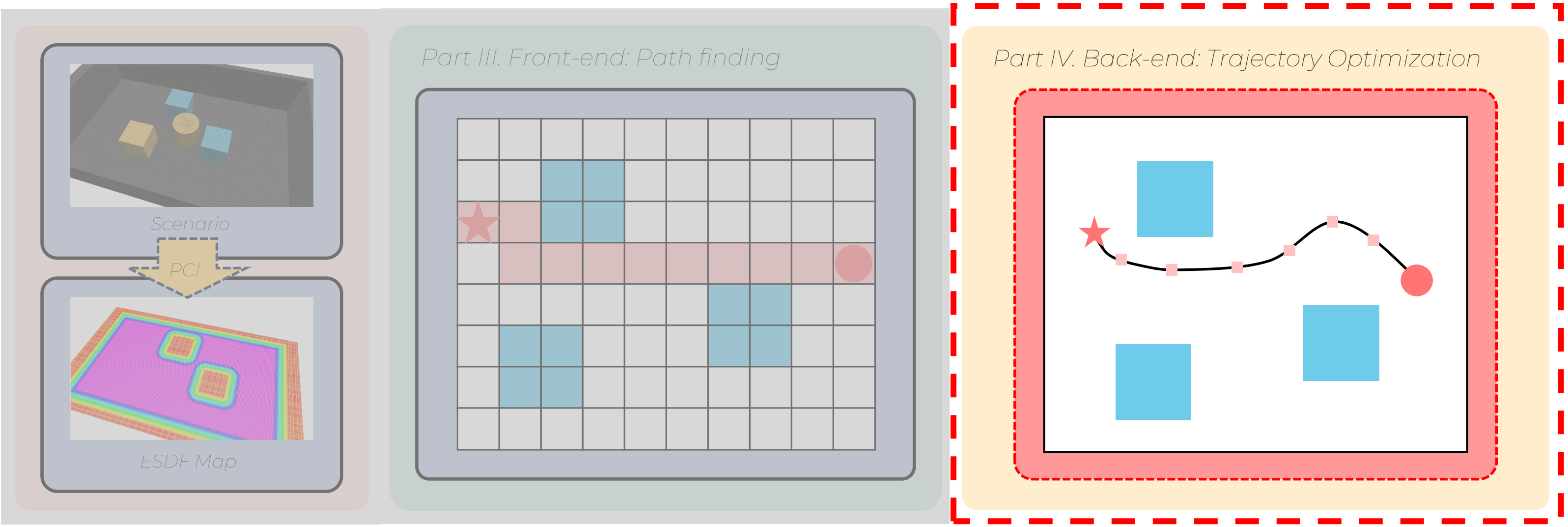




Smooth trajectory is beneficial for mobile robots:

- Smooth trajectories respect the continuous nature of robots.
- The mobile robot should not stop at turns.

[1] L. Quan, L. Han, B. Zhou, S. Shen, and F. Gao, "Survey of uav motion planning," IET Cyber-Systems and Robotics, vol. 2, no. 1, pp. 14–21, 2020. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-csr.2020.0004



Trajectory Optimization

- Problem Formulization
- Hard-Constrained Trajectories Generation
- Spatial-Temporal Optimization

Problem Formulization

- Trajectory representation

Polynomials

$$\mathbf{f}_p(t) : [x(t), y(t), z(t)]^T$$

where $q_x(t), q_y(t),$ and $q_z(t)$ are three real-valued polynomial functions.

$$q_x(t) = \sum_{i=0}^{M} \mathbf{c}_i^T \eta_i(t),\ t \in [0, T]$$

where $\mathbf{c}_i = [c_0, c_1, \cdots, c_{n-1}]^T$ is a coefficient vector, and
$\eta_i(t) = \left[1, t, t^2, \cdots, t^{n-1}\right]^T$ is a base function.

Bézier curves / B-spline

$$x(t) = \sum_{i=0}^{M} p_i \cdot B_{M,i}(t)$$

Problem Formulization

HKUST

- Optimization problem with equality and inequality constraints[1]:

$$\min_{x \in \mathcal{D}} f(x)$$

$$\text{s.t.} \quad g_i(x) \leq 0 \qquad \forall i \in \{1, \ldots, m_g\}$$

$$h_j(x) = 0 \qquad \forall j \in \{1, \ldots, m_h\}$$

where f, g_i, and h_j are all smooth, real-valued functions defined on $\mathbb{R}^n$. We call f the *objective* function, while $g_i, i \in \mathcal{L}$ are the *inequality constraints* and $h_j, j \in \mathcal{E}$ are the *equality constraints*. The feasible set Ω is the set of points x that satisfy the constraints, i.e.

$$\Omega = \{x \in \mathcal{D} \mid g_i(x) \leq 0 \; \forall i \in \mathcal{L},\; h_j(x) = 0 \; \forall j \in \mathcal{E}\}$$

$$\min_{t_0, t_F, \boldsymbol{x}(t), \boldsymbol{u}(t)} \underbrace{J(t_0, t_F, x(t_0), x(t_F))}_{\text{Mayer Term}} + \underbrace{\int_{t_0}^{t_F} w(\tau, x(\tau), u(\tau))\, d\tau}_{\text{Lagrange Term}}$$

$\dot{x}(t) = f(t, x(t), u(t))$ **system dynamics**

$h(t, x(t), u(t)) \leq 0$ **path constraint**

$g(t_0, t_F, x(t_0), x(t_F)) \leq 0$ **boundary constraint**

$x_{\text{low}} \leq x(t) \leq x_{\text{upp}}$ **path bound on state,**

$u_{\text{low}} \leq u(t) \leq u_{\text{upp}}$ **path bound on control.**

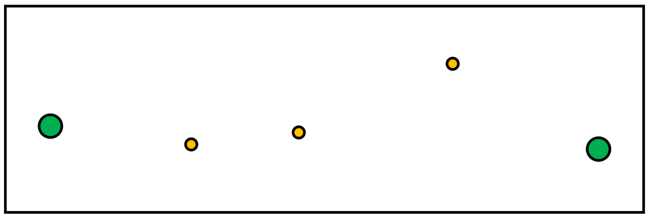
$t_{\text{low}} \leq t_0 \leq t_F \leq t_{\text{upp}}$ **bounds on initial and final time,**

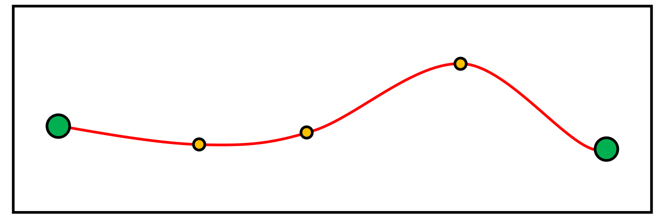
$x_{0,\text{low}} \leq x(t_0) \leq x_{0,\text{upp}}$ **bound on initial state,**

$x_{F,\text{low}} \leq x(t_F) \leq x_{F,\text{upp}}$ **bound on final state.**

[1] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017.

Hard-Constrained Trajectories





$$\mathbf{p} = [p_0, p_1, \cdots p_M]^T$$

Input: waypoints

$$\mathbf{C} = [\mathbf{c}_0, \mathbf{c}_1, \cdots, \mathbf{c}_{M-1}]^T$$

Output: smooth trajectory

- Example ($M = 1$) $\quad x(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 + c_5 t^5$

	Position	Velocity	Acceleration
$t = 0$	p_0	0	0
$t = T$	p_M	0	0

	Position	Velocity	Acceleration
$t = 0$	p_0	v_0	0
$t = T$	p_M	v_T	0

$$\begin{bmatrix} p_0 \\ p_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix}$$

$$\begin{bmatrix} p_0 \\ p_1 \\ v_0 \\ v_T \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix}$$

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L4_spring_2023.pdf

Hard-Constrained Trajectories

- Example (M segments)

$$x(t)\begin{cases} x_1(t)=\sum_{i=0}^{n}\mathbf{c}_{1,i}\eta_{1,i}(t) & t\in[0,T_1] \\ x_2(t)=\sum_{i=0}^{n}\mathbf{c}_{2,i}\eta_{2,i}(t) & t\in[T_1,T_2] \\ \vdots & \vdots \\ x_M(t)=\sum_{i=0}^{n}\mathbf{c}_{M,i}\eta_{M,i}(t) & t\in[T_{M-1},T_M] \end{cases}$$

$$\mathbf{M}(\mathbf{T})\cdot\mathbf{c}=\mathbf{b}(\mathbf{p})$$

Find $\mathbf{c}^*_{\mathbf{p},\mathbf{T}}$

Subject to:

$$\begin{cases} x_j^{(k)}(T_{j-1})=x_{j,0}^{(k)} \\ x_j^{(k)}(T_j)=x_{j,T}^{(k)} \end{cases}$$

Derivative constraints

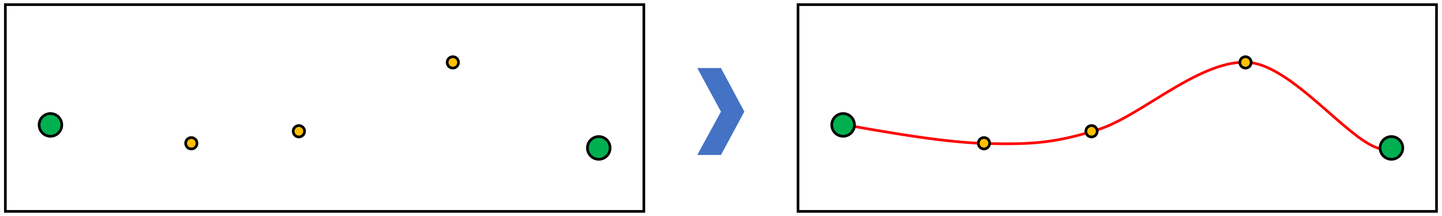
$$x_j^{(k)}(T_j)=x_{j+1}^{(k)}(T_j)$$

Continuity constraints

Cost function

$$\mathcal{J}(T)=\int_0^T\left(x^{(k)}(t)\right)^2dt=\begin{bmatrix}\vdots\\ c_i\\ \vdots\end{bmatrix}^T\begin{bmatrix} & & \vdots & & \\ \cdots & & \frac{i\cdot(i-1)\cdots(i-k+1)\cdot j\cdot(j-1)\cdots(j-k+1)}{i+j-2k+1}T^{i+j-2k+1} & & \cdots \\ & & \vdots & & \end{bmatrix}\begin{bmatrix}\vdots\\ c_j\\ \vdots\end{bmatrix}$$

Hard-Constrained Trajectories



$\mathbf{p} = [p_0, p_1, \cdots p_M]^T$

Input: waypoints

$\mathbf{C} = [\mathbf{c}_0, \mathbf{c}_1, \cdots, \mathbf{c}_{M-1}]^T$

Output: smooth trajectory

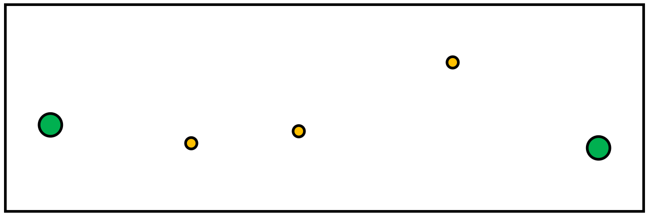
Find $\mathbf{c}^*_{\mathbf{p},\mathbf{T}}$ $\quad \min \mathcal{J}(T) = \mathbf{c}^{\mathbf{T}}\mathbf{Q}\mathbf{c}$

s.t. $\mathbf{M}(\mathbf{T}) \cdot \mathbf{c} = \mathbf{b}(\mathbf{p})$

- ✓ **Constrained quadratic programming (QP) formulation.**
- ✓ **Continuous trajectory with finite parameters.**

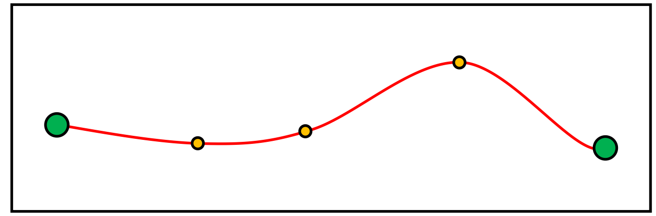
- ☐ **How to ensure collision-free trajectories?**
- ☐ **How to determine the time allocation for each segment?**

Hard-Constrained Trajectories



$$\mathbf{p} = [p_0, p_1, \cdots p_M]^T$$

Input: waypoints



$$\mathbf{C} = [\mathbf{c}_0, \mathbf{c}_1, \cdots, \mathbf{c}_{M-1}]^T$$

Output: smooth trajectory

Find $\mathbf{c}^*_{\mathbf{p},\mathbf{T}}$ $\quad \min \mathcal{J}(T) = \mathbf{c}^{\mathbf{T}}\mathbf{Q}\mathbf{c}$

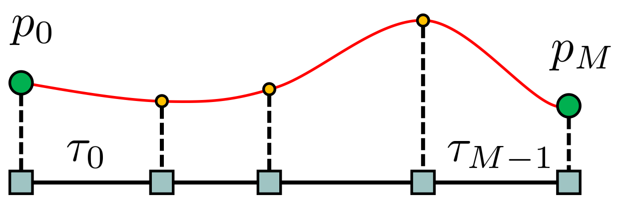
s.t. $\mathbf{M}(\mathbf{T}) \cdot \mathbf{c} = \mathbf{b}(\mathbf{p})$

- ✓ **Constrained quadratic programming (QP) formulation.**
- ✓ **Continuous trajectory with finite parameters.**

We need spatial-temporal optimization!

Spatial-Temporal Optimization

STEP 1



$$\mathbf{x} = [\underbrace{p_0, \ldots, p_M}_{\mathbf{p}}, \underbrace{\tau_0, \ldots, \tau_{M-1}}_{\mathbf{T}}]^T$$

Decision variables: waypoints and time allocation

$$\min \mathcal{J}(T) = \mathbf{c}^{\mathbf{T}}\mathbf{Q}\mathbf{c}$$

$$\mathbf{M}(\mathbf{T}) \cdot \mathbf{c} = \mathbf{b}(\mathbf{p}) \rightarrow \mathbf{c}_{\mathbf{x}}$$

A general trajectory optimization problem:

$$\min_{t_0, t_F, \boldsymbol{x}(t), \boldsymbol{u}(t)} \underbrace{J(t_0, t_F, x(t_0), x(t_F))}_{\text{Mayer Term}} + \underbrace{\int_{t_0}^{t_F} w(\tau, x(\tau), u(\tau))\, d\tau}_{\text{Lagrange Term}}$$

$\dot{x}(t) = f(t, x(t), u(t))$	**system dynamics**
$h(t, x(t), u(t)) \leq 0$	**path constraint**
$g(t_0, t_F, x(t_0), x(t_F)) \leq 0$	**boundary constraint**
$x_{\text{low}} \leq x(t) \leq x_{\text{upp}}$	**path bound on state,**
$u_{\text{low}} \leq u(t) \leq u_{\text{upp}}$	**path bound on control.**
$t_{\text{low}} \leq t_0 \leq t_F \leq t_{\text{upp}}$	**bounds on initial and final time,**
$x_{0,\text{low}} \leq x(t_0) \leq x_{0,\text{upp}}$	**bound on initial state,**
$x_{F,\text{low}} \leq x(t_F) \leq x_{F,\text{upp}}$	**bound on final state.**

Spatial-Temporal Optimization

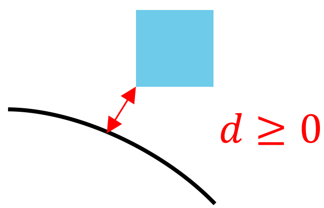
STEP 2

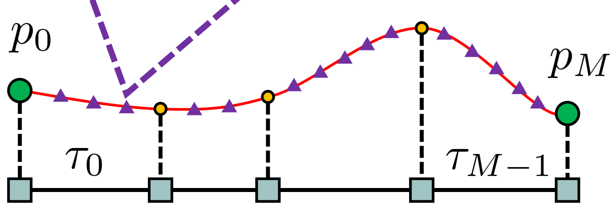
$$\min_{\mathbf{x}=[\mathbf{q},\mathbf{T}]^T} \mathcal{F}(\mathbf{x}) = \underbrace{\int_0^{\|\mathbf{T}\|} \mathbf{q}^{(s)}(\mathbf{c_x},t)^T \mathbf{Q}\mathbf{q}^{(s)}(\mathbf{c_x},t)dt}_{\text{Lagrange term}} + \underbrace{\alpha\|\mathbf{T}\|}_{\text{Mayer term}}$$

Polynomial

$$\mathbf{q}(\mathbf{c_x},t) = \sum_{i=0}^{M} \mathbf{c}_i^T \eta_i(t) \qquad \mathbf{q}^{[s]} = \left[\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \cdots, \mathbf{q}^{(s)}\right]^T$$

$$\text{subject to} \quad \mathbf{g}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) \leq \mathbf{0}, \quad \mathbf{h}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) = \mathbf{0}.$$





A general trajectory optimization problem:

$$\min_{t_0,t_F,\boldsymbol{x}(t),\boldsymbol{u}(t)} \underbrace{J(t_0,t_F,x(t_0),x(t_F))}_{\text{Mayer Term}} + \underbrace{\int_{t_0}^{t_F} w\big(\tau,x(\tau),u(\tau)\big)\,d\tau}_{\text{Lagrange Term}}$$

$\dot{x}(t) = f(t,x(t),u(t))$	**system dynamics**
$h(t,x(t),u(t)) \leq 0$	**path constraint**
$g(t_0,t_F,x(t_0),x(t_F)) \leq 0$	**boundary constraint**
$x_{\text{low}} \leq x(t) \leq x_{\text{upp}}$	**path bound on state,**
$u_{\text{low}} \leq u(t) \leq u_{\text{upp}}$	**path bound on control.**
$t_{\text{low}} \leq t_0 \leq t_F \leq t_{\text{upp}}$	**bounds on initial and final time,**
$x_{0,\text{low}} \leq x(t_0) \leq x_{0,\text{upp}}$	**bound on initial state,**
$x_{F,\text{low}} \leq x(t_F) \leq x_{F,\text{upp}}$	**bound on final state.**

Spatial-Temporal Optimization

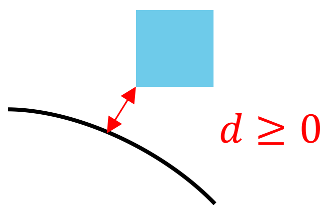
STEP 2

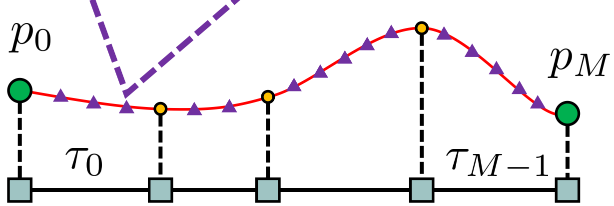
$$\min_{\mathbf{x}=[\mathbf{q},\mathbf{T}]^T} \mathcal{F}(\mathbf{x}) = \underbrace{\int_0^{\|\mathbf{T}\|} \mathbf{q}^{(s)}(\mathbf{c_x},t)^T \mathbf{Q}\mathbf{q}^{(s)}(\mathbf{c_x},t)dt +}_{\text{Lagrange term}} \underbrace{\alpha\|\mathbf{T}\|}_{\text{Mayer term}}$$

Polynomial

$$\mathbf{q}(\mathbf{c_x},t) = \sum_{i=0}^{M} \mathbf{c}_i^T \eta_i(t) \qquad \mathbf{q}^{[s]} = \left[\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \cdots, \mathbf{q}^{(s)}\right]^T$$

$$\text{subject to} \quad \mathbf{g}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) \leq \mathbf{0}, \qquad \mathbf{h}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) = \mathbf{0}.$$





STEP 3

$$\frac{\partial \mathcal{F}}{\partial \mathbf{x}} = \frac{\partial \mathcal{F}}{\partial \mathbf{q}} \cdot \frac{\partial \mathbf{q}}{\partial \mathbf{c}} \cdot \left(\frac{\partial \mathbf{c}}{\partial \mathbf{x}} + \frac{\partial \mathbf{c}}{\partial \mathbf{T}}\right) + \frac{\partial \mathcal{F}}{\partial \mathbf{T}}$$

Acceleration skills

$$\mathbf{x}^* = \arg\min_{\mathbf{x}=[\mathbf{q},\mathbf{T}]^T} \mathcal{F}(\mathbf{x}) + \boxed{\mathcal{L}_{\mu,\lambda,\rho}(\mathbf{x})}$$

$$\boxed{\begin{aligned} &\mathbf{g}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) \leq \mathbf{0}, \\ &\mathbf{h}\left(\mathbf{q}^{[s]}(\mathbf{c_x})\right) = \mathbf{0}. \end{aligned}}$$

Unconstrained optimization

a. Scaling objective function & constraints

b. L-BFGS method

c. Lewis-Overton method

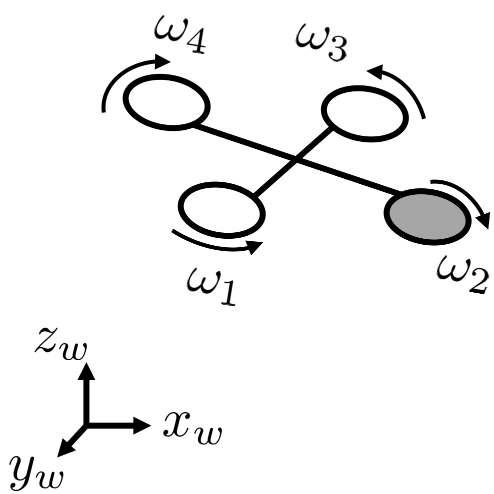
Applications

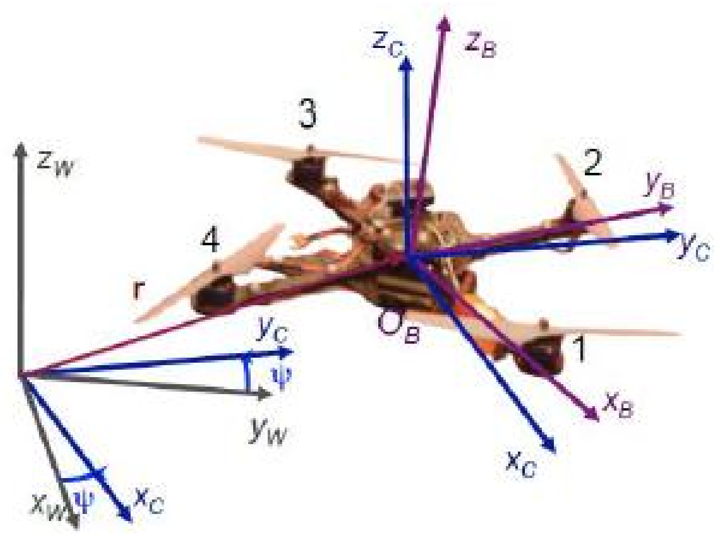
- Differential Flatness
- Minimum Snap Trajectory
- A Quadrotor with A Cable-Suspended Payload
- Navigation Among Movable Objects

A single quadrotor's motion planning

Contact-aware motion planning

Differential Flat Output





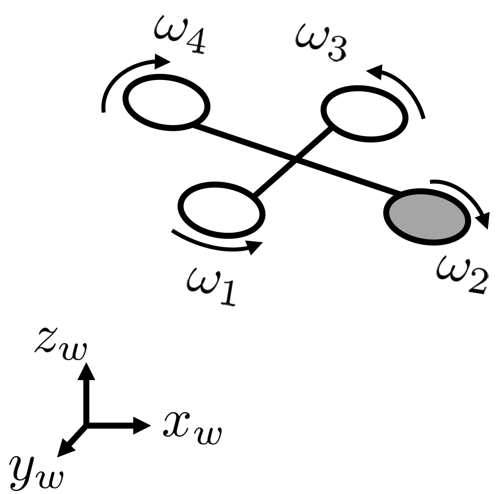
- Quadrotor states: position, orientation, linear velocity, and angular velocity.

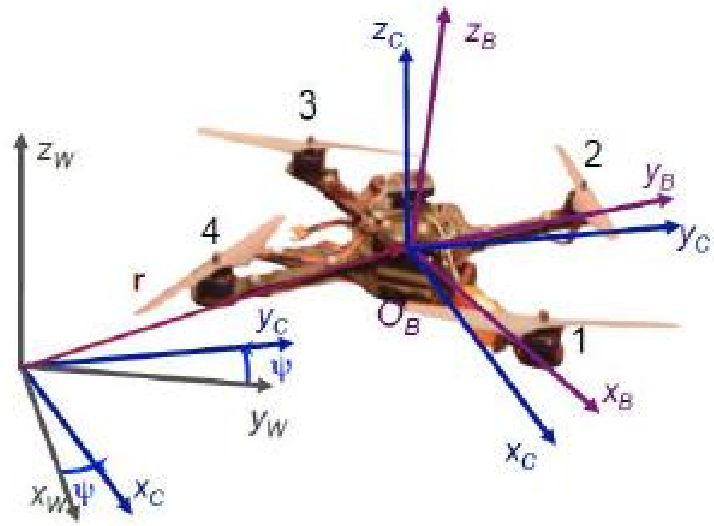
$$\mathbf{x} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \boxed{\omega_x, \omega_y, \omega_z}]^T$$

Body angular velocity viewed in the body frame.

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L3_spring_2023.pdf

Differential Flat Output





- Quadrotor states: position, orientation, linear velocity, and angular velocity.

$$\mathbf{x} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \boxed{\omega_x, \omega_y, \omega_z}]^T$$

Body angular velocity viewed in the body frame.

- The states and the inputs of a quadrotor can be written as algebraic functions of four carefully selected flat outputs and their derivatives.

$$\sigma = [x, y, z, \psi]^T \qquad [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z] = f\left(\sigma^{[s]}\right)$$

[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L3_spring_2023.pdf

Differential Flat Output

- The states and the inputs of a quadrotor can be written as algebraic functions of four carefully selected flat outputs and their derivatives.

$$\sigma = [x, y, z, \psi]^T \qquad [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z] = f\left(\sigma^{[s]}\right)$$

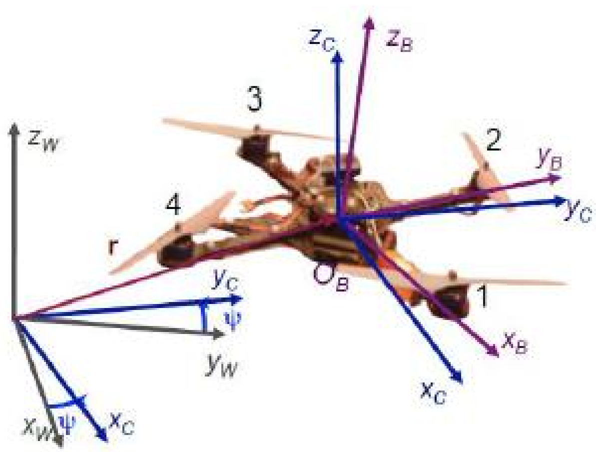
Observations

- Position, velocity, and acceleration are simply derivatives of the flat outputs.
- From the equation of motion, we can calculate the z-axis of body directly:

$$\mathbf{z}_B = \frac{\mathbf{t}}{\|\mathbf{t}\|}, \ \mathbf{t} = [\ddot{x}, \ddot{y}, \ddot{z} + g]^T.$$

- Given the yaw vector: $\mathbf{x}_C = [\cos\psi, \sin\psi, 0]^T$
- The orientation can be expressed in terms of flat outputs:

$$\mathbf{y}_B = \frac{\mathbf{z}_B \times \mathbf{x}_C}{\|\mathbf{z}_B \times \mathbf{x}_C\|}, \ \mathbf{x}_B = \mathbf{y}_B \times \mathbf{z}_B, \ \mathbf{R}_B = [\mathbf{x}_B \ \mathbf{y}_B \ \mathbf{z}_B]$$



[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L3_spring_2023.pdf

Differential Flat Output

- The states and the inputs of a quadrotor can be written as algebraic functions of four carefully selected flat outputs and their derivatives.

$$\sigma = [x, y, z, \psi]^T \qquad [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z] = f\left(\sigma^{[s]}\right)$$

Observations

- Take the derivative of the equation of motion:

$$m\ddot{\mathbf{p}} = -mg\mathbf{z}_W + f\mathbf{z}_B \Longrightarrow m\mathbf{p}^{(3)} = \dot{f}\mathbf{z}_B +^W \omega_B \times f\mathbf{z}_B$$

- Quadrotors only have vertical thrust: $\dot{f} = \mathbf{z}_B \cdot m\mathbf{p}^{(3)}$
- If we define:

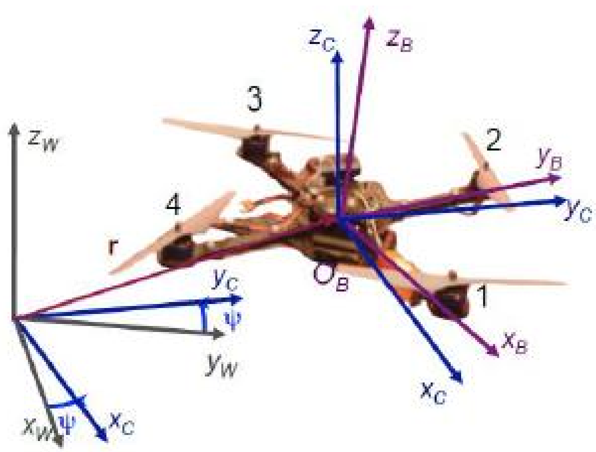
$$\mathbf{h}_\omega =^W \omega_B \times \mathbf{z}_B = \frac{m}{f}\left(\mathbf{p}^{(3)} - \left(\mathbf{z}_B \cdot \mathbf{p}^{(3)}\right)\mathbf{z}_B\right)$$

- We know that:

$$^W\omega_B = \omega_x \mathbf{x}_B + \omega_y \mathbf{y}_B + \omega_z \mathbf{z}_B$$

- Angular velocities in body frame can be found as:

$$\omega_x = -\mathbf{h}_\omega \cdot \mathbf{y}_B,\ \omega_y = \mathbf{h}_\omega \cdot \mathbf{x}_B,\ \omega_z = \dot{\psi}\mathbf{z}_W \cdot \mathbf{z}_B$$



[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L3_spring_2023.pdf

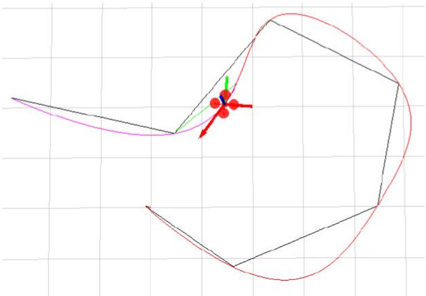
Minimum Snap Trajectory

➤ The states and the inputs of a quadrotor can be written as algebraic functions of four carefully selected flat outputs and their derivatives.

$$\sigma = [x, y, z, \psi]^T \qquad [\phi, \theta, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z] = f\left(\sigma^{[s]}\right)$$

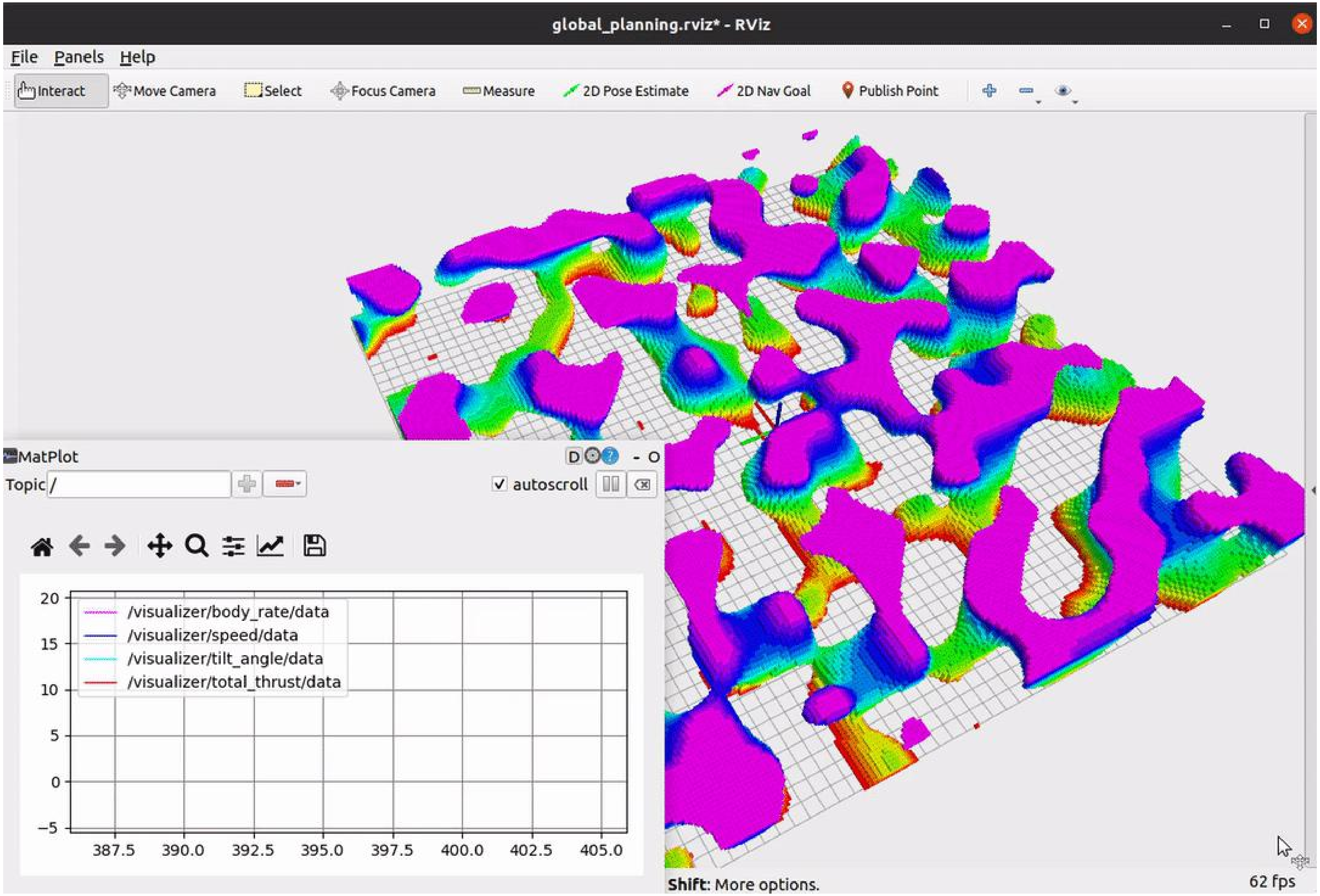
Derivative	Translation	Orientation	Thrust
σ	Position	-	-
$\sigma^{(1)}$	Velocity	-	-
$\sigma^{(2)}$	Acceleration	Rotation	-
$\sigma^{(3)}$	Jerk	Angular velocity	-
$\sigma^{(4)}$	Snap	Angular acceleration	Differential thrust

Minimum snap: $\mathcal{J}(T) = \int_0^T \left(\sigma^{(4)}(t)\right)^2 dt$



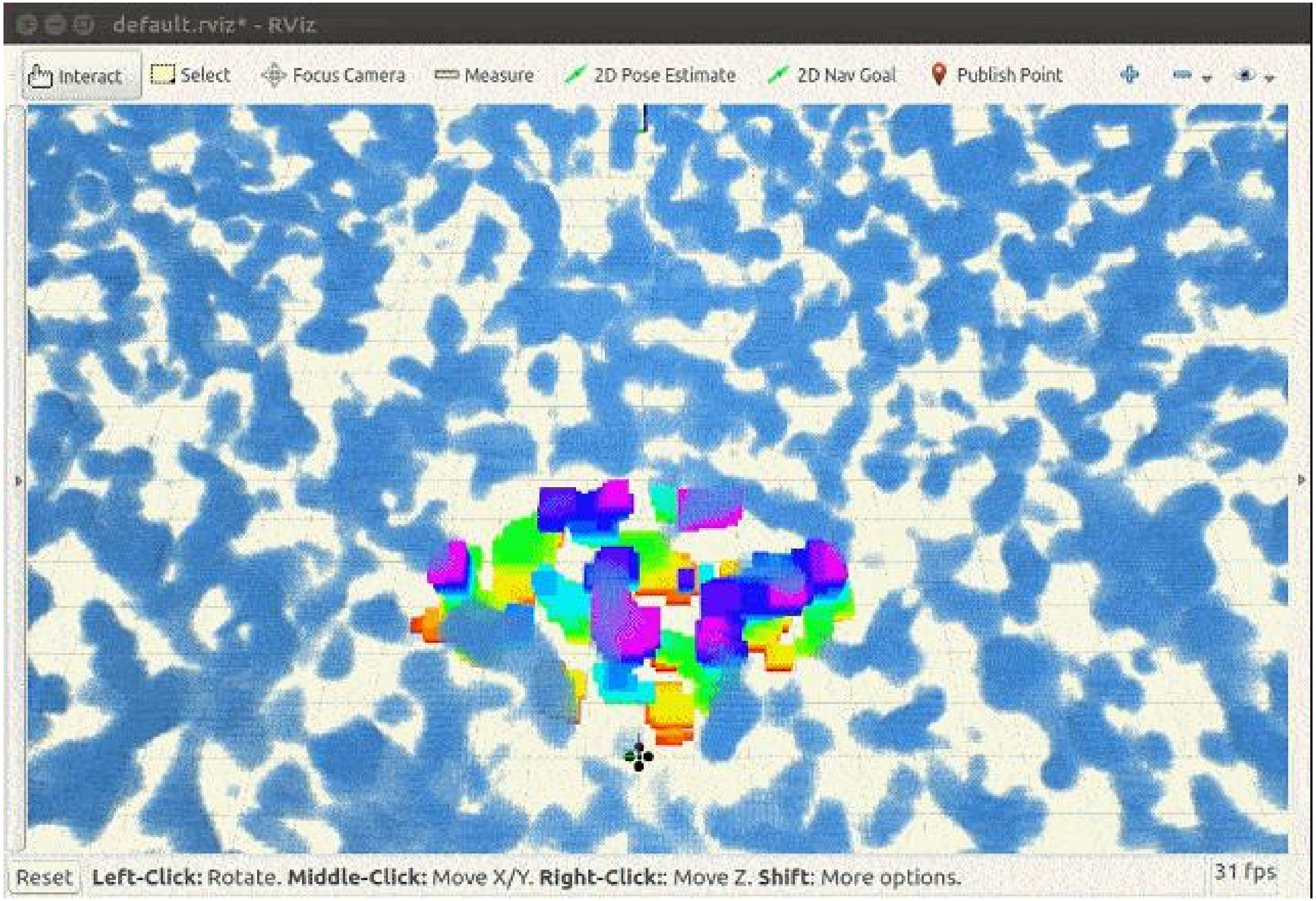
[1] Reference: https://github.com/HKUST-Aerial-Robotics/HKUST-ELEC5660-Introduction-to-Aerial-Robots/blob/ver_2023/course_note/L3_spring_2023.pdf

Global Planning of A Quadrotor



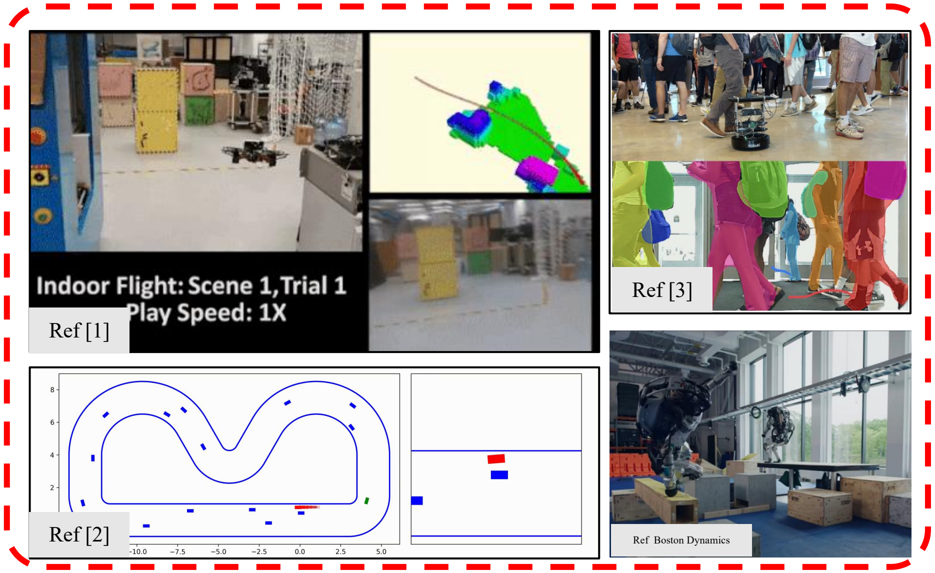
[1] Reference: https://github.com/ZJU-FAST-Lab/GCOPTER

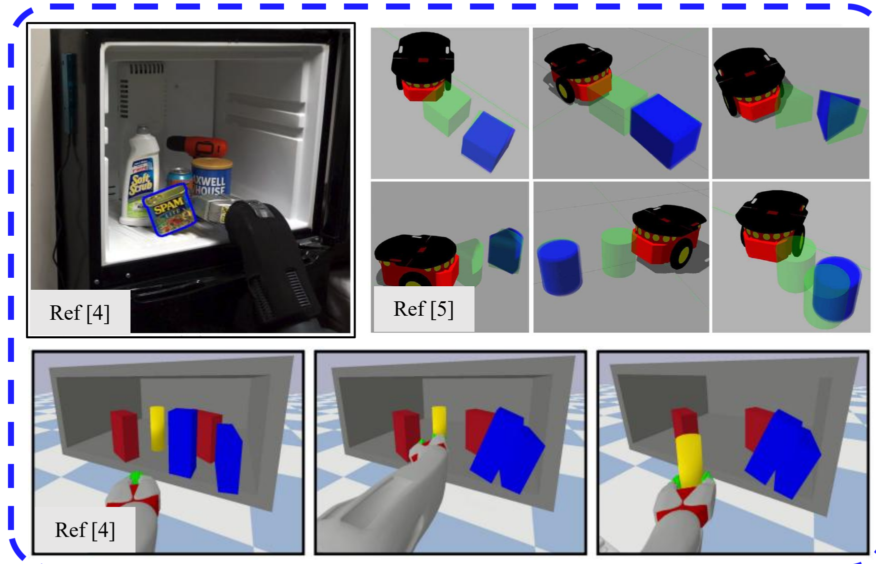
Local Replanning



[1] Reference: https://github.com/ZJU-FAST-Lab/ego-planner

Complex Scenarios





Interaction:
- Obstacle avoidance
- Contact-aware
- …

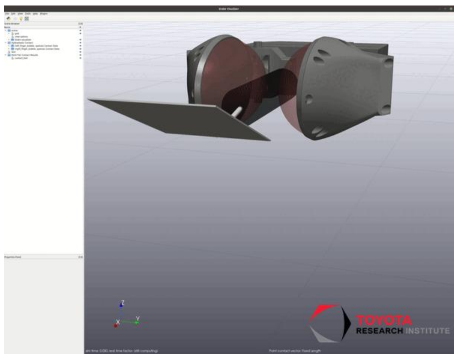
System dynamics:
- Physical limitations
- Feasibility
- Safety
- …

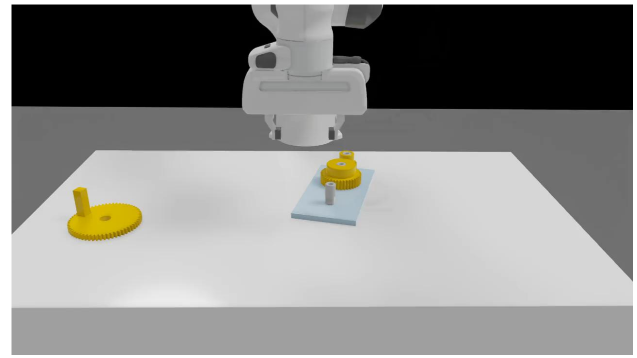
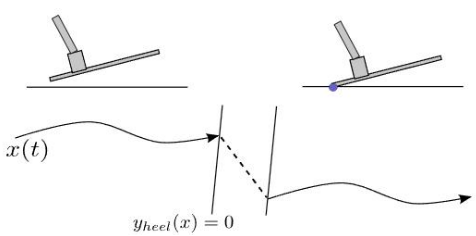
Mission goals:
- Navigation
- Pick-and-place
- Non-prehensile manipulation
- …

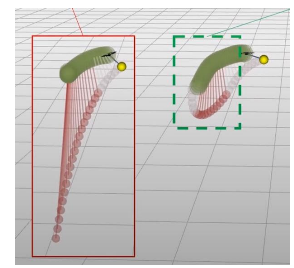
[1] B. Zhou, F. Gao, L. Wang, C. Liu and S. Shen, "Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight," in IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 3529-3536, Oct. 2019, doi: 10.1109/LRA.2019.2927938.
[2] S. He, J. Zeng and K. Sreenath, "Autonomous Racing with Multiple Vehicles using a Parallelized Optimization with Safety Guarantee using Control Barrier Functions," 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 3444-3451, doi: 10.1109/ICRA46639.2022.9811969.
[3] A. J. Sathyamoorthy, J. Liang, U. Patel, T. Guan, R. Chandra and D. Manocha, "DenseCAvoid: Real-time Navigation in Dense Crowds using Anticipatory Behaviors," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 11345-11352, doi: 10.1109/ICRA40945.2020.9197379.
[4] D. M. Saxena, and M. Likhachev, "Planning for Complex Non-prehensile Manipulation Among Movable Objects by Interleaving Multi-Agent Pathfinding and Physics-Based Simulation, " 2023 arXiv preprint arXiv:2303.13352.
[5] J. St¨uber, C. Zito, and R. Stolkin, “Let’s push things forward: A survey on robot pushing,” Frontiers in Robotics and AI, vol. 7, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2020.00008

Complex Scenarios

Motivation: Demand for Contact-Aware Motion Planning



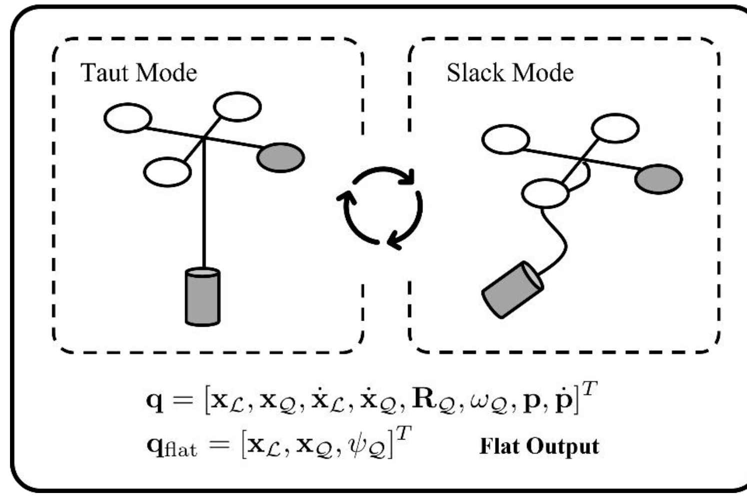



Modes Switching

Make / Brake Collision

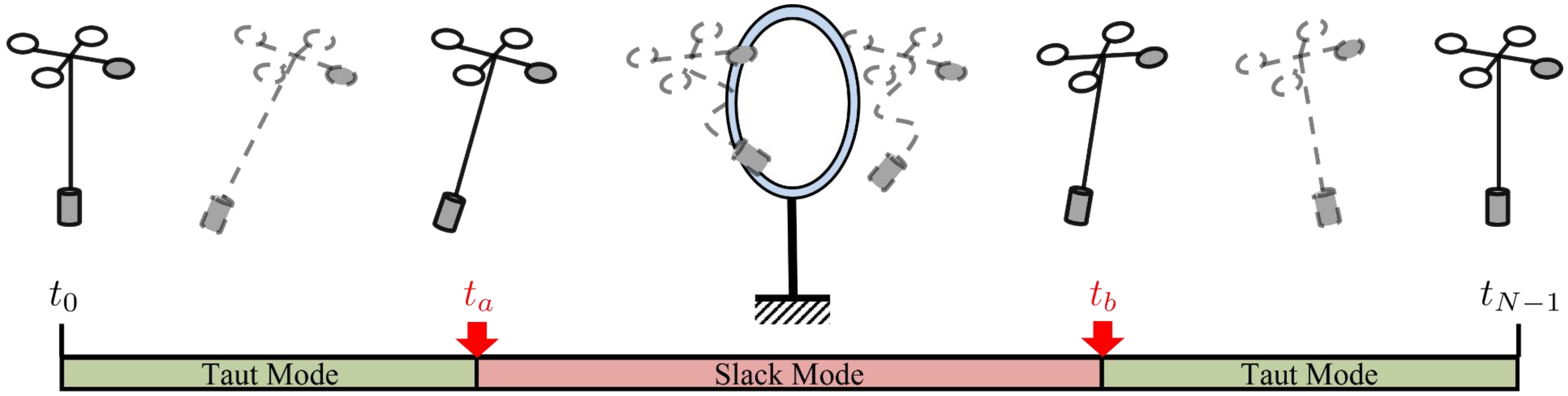
[1] M. Sherman, “Rethinking contact simulation for robot manipulation,” Medium, https://medium.com/toyotaresearch/rethinking-contact-simulation-for-robot-manipulation-434a56b5ec88 (accessed Oct. 27, 2023).
[2] Russ Tedrake. Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832).

A Quadrotor with A Cable-Suspended Payload



- Aerial robots with suspended payloads play a vital role in logistics, transportation, and disaster relief scenarios.
- One intriguing aspect of this system is that it exhibits dual motion modes in agile flights. It is attractive to exploit mode switching for the suspended payload system, which allows for performing transportation tasks in complex and narrow spaces.

A Quadrotor with A Cable-Suspended Payload



- The existence of dual system dynamics descriptions dramatically increases the complexity of feasible planning and practical control.
- Potential impacts caused by transitions between two subsystems will break the robotic status consistency.

A Quadrotor with A Cable-Suspended Payload

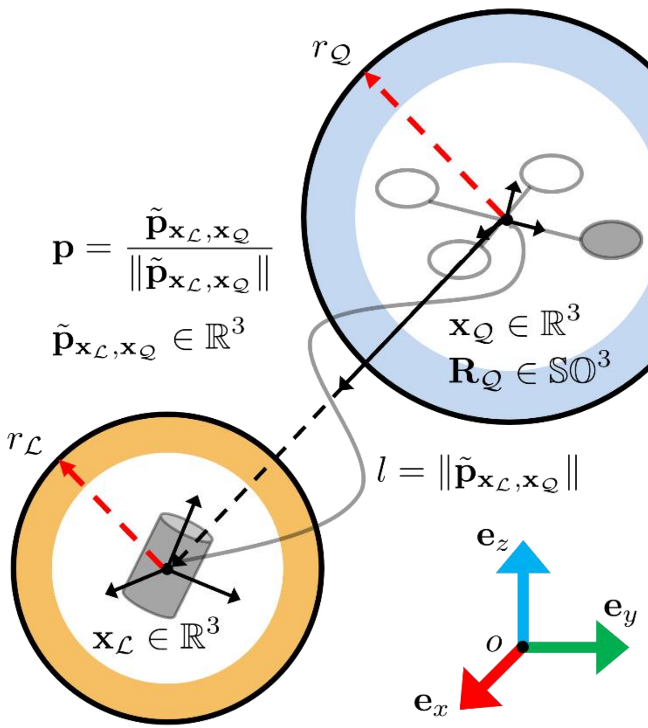
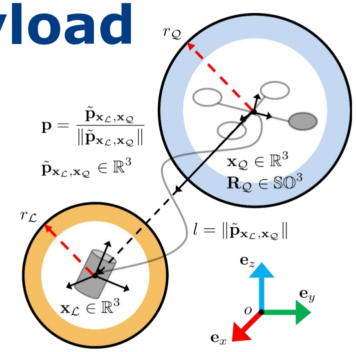


TABLE I

SYMBOLS USED IN THIS PAPER

$\mathcal{Q}, \mathcal{L}, \mathcal{W}$	Quadrotor frame, payload frame, and world frame.
$\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z \in \mathbb{R}^3$	Unit vector in the axis-direction of the world frame.
$\mathbf{x}_{\mathcal{B}}, \mathbf{y}_{\mathcal{B}}, \mathbf{z}_{\mathcal{B}} \in \mathbb{R}^3$	Unit vector in the axis-direction of the quadrotor frame.
$\mathbf{x}_{\mathcal{Q}}, \mathbf{x}_{\mathcal{L}} \in \mathbb{R}^3$	Position of the quadrotor and the payload.
$\mathbf{v}_{\mathcal{Q}}, \mathbf{v}_{\mathcal{L}} \in \mathbb{R}^3$	Linear velocity of quadrotor and the payload.
$\mathbf{R}_{\mathcal{Q}} \in \mathbb{R}^3$	Orientation of the quadrotor.
$\omega_{\mathcal{Q}} \in \mathbb{R}^3$	Body rate of the quadrotor.
$m_{\mathcal{Q}}, m_{\mathcal{L}} \in \mathbb{R}$	Mass of the quadrotor and the payload.
$r_{\mathcal{Q}}, r_{\mathcal{L}} \in \mathbb{R}$	Safe margin diameter of the quadrotor and the payload.
$\tilde{\mathbf{p}} \in \mathbb{R}^3$	Vector pointing from the quadrotor to the payload.
$\mathbf{p} \in \mathbb{S}^2$	The unit vector in the direct of $\tilde{\mathbf{p}}$.
$\dot{\mathbf{p}} \in \mathbb{R}^3$	The first derivative with respect to time of vector $\mathbf{p}$.
$l \in \mathbb{R}$	The distance between the quadrotor and the payload.
$l_0 \in \mathbb{R}$	The cable length.
$f,\ f_T \in \mathbb{R}$	Thrust of the quadrotor, and tension in the cable.

A Quadrotor with A Cable-Suspended Payload



$$f_T(t) \geq 0, \quad (1)$$

$$l_0 - l(t) \geq 0, \quad (2)$$

$$f_T(t)(l_0 - l(t)) = 0, \quad (3)$$

Complementarity constraints

$$f_T = \|m_\mathcal{L}(\ddot{\mathbf{x}}_\mathcal{L} + g\mathbf{e}_z)\|, \quad (4)$$

$$\tilde{\mathbf{p}} = \mathbf{x}_\mathcal{L} - \mathbf{x}_Q, \quad (5)$$

$$l = \|\tilde{\mathbf{p}}\| = \|\mathbf{x}_\mathcal{L} - \mathbf{x}_Q\|, \quad (6)$$

$$\mathbf{p} = \frac{\tilde{\mathbf{p}}}{\|\tilde{\mathbf{p}}\|}, \quad (7)$$

Definition of distance and direction

$$\mathbf{z}_\mathcal{B} = \frac{m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}}{\|m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}\|} = \frac{1}{f}(m_Q(\ddot{\mathbf{x}}_Q + g\mathbf{e}_z) - f_T\mathbf{p}), \quad (8)$$

$$\mathbf{y}_\mathcal{B} = \frac{\mathbf{z}_\mathcal{B} \times \mathbf{x}_\mathcal{C}}{\|\mathbf{z}_\mathcal{B} \times \mathbf{x}_\mathcal{C}\|}, \text{ where } \mathbf{x}_\mathcal{C} = [\cos\psi, \sin\psi, 0]^T, \quad (9)$$

$$\mathbf{x}_\mathcal{B} = \mathbf{y}_\mathcal{B} \times \mathbf{z}_\mathcal{B}. \quad (10)$$

Calculation of quadrotor's orientation

Contact-Aware Motion Planning

- <u>O</u>ptimization with <u>n</u>onlinear <u>c</u>omplementarity <u>c</u>onstraints (ONCC):

$$\min_{\mathbf{x}} \quad \underbrace{\int_0^{\|\mathbf{T}\|_1} \mathbf{q}_{\mathbf{x}}^{(k)^T}(\tau)\mathbf{Q}_w \mathbf{q}_{\mathbf{x}}^{(k)}(\tau)\mathrm{d}\tau}_{\text{Lagrange term}} + \underbrace{\alpha\|\mathbf{T}\|_1}_{\text{Mayer term}} \tag{14a}$$

$$\text{subject to} \quad \mathbf{g}(\mathbf{q}_{\mathbf{x}}^{[k]}(t)) \leq \mathbf{0}, \tag{14b}$$

$$\mathbf{h}(\mathbf{q}_{\mathbf{x}}^{[k]}(t)) = \mathbf{0}, \tag{14c}$$

$$\forall t \in [0, \|\mathbf{T}\|_1], \tag{14d}$$

where

$$\mathbf{q}_{\mathbf{x}}^{[k]}(t) = [\mathbf{q}_{\mathbf{x}}^T(t), \dot{\mathbf{q}}_{\mathbf{x}}^T(t), \cdots, \mathbf{q}_{\mathbf{x}}^{(k)^T}(t)]^T$$

$$\mathbf{q}_{i,\mathbf{x}}(t) = \mathbf{c}_{i,\mathbf{x}}\phi(t - t_{i-1}), \; t \in [t_{i-1}, t_i]$$

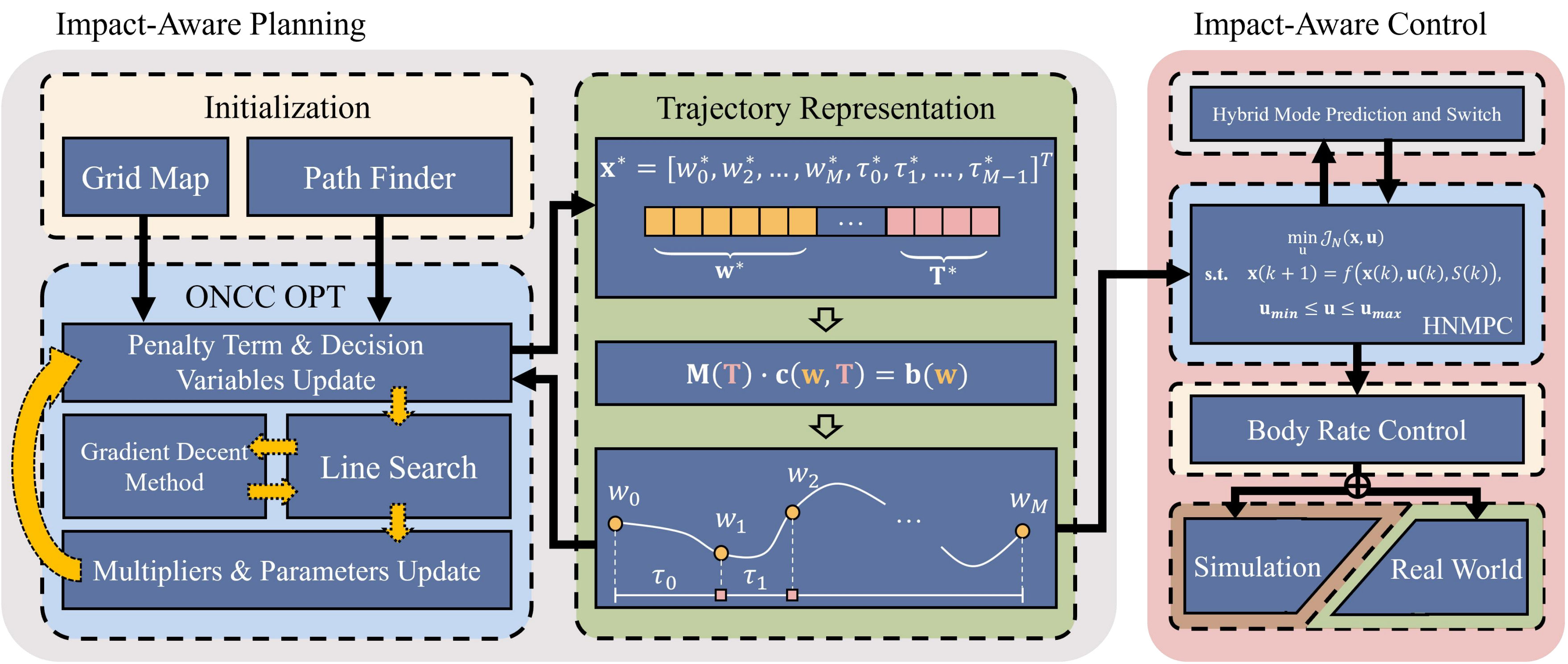
$$\mathbf{c}_{i,\mathbf{x}} \in \mathbf{R}^{N \times 2k}$$

$$\phi(a) = [1, a, a^2, \cdots, a^{2k-1}]^T$$

Q: How to solve such problem efficient?

- **The optimal energy polynomials: MINCO trajectories**
- **Efficient optimization framework: PHR-ALM**

Contact-Aware Motion Planning



Contact-Aware Motion Planning

$$\min_{t_0, t_F, x(t), u(t)} \underbrace{J(t_0, t_F, x(t_0), x(t_F))}_{\text{Mayer Term}} + \underbrace{\int_{t_0}^{t_F} w(\tau, x(\tau), u(\tau))\, d\tau}_{\text{Lagrange Term}}$$

$\dot{x}(t) = f(t, x(t), u(t))$	**system dynamics**
$h(t, x(t), u(t)) \leq 0$	**path constraint**
$g(t_0, t_F, x(t_0), x(t_F)) \leq 0$	**boundary constraint**
$x_{\text{low}} \leq x(t) \leq x_{\text{upp}}$	**path bound on state,**
$u_{\text{low}} \leq u(t) \leq u_{\text{upp}}$	**path bound on control.**
$t_{\text{low}} \leq t_0 \leq t_F \leq t_{\text{upp}}$	**bounds on initial and final time,**
$x_{0,\text{low}} \leq x(t_0) \leq x_{0,\text{upp}}$	**bound on initial state,**
$x_{F,\text{low}} \leq x(t_F) \leq x_{F,\text{upp}}$	**bound on final state.**

$$\min_{\mathbf{x}} \underbrace{\int_0^{\|\mathbf{T}\|_1} \mathbf{q}_{\mathbf{x}}^{(k)^T}(\tau)\mathbf{Q}_w \mathbf{q}_{\mathbf{x}}^{(k)}(\tau)d\tau}_{\text{Lagrange term}} + \underbrace{\alpha\|\mathbf{T}\|_1}_{\text{Mayer term}} \quad (14a)$$

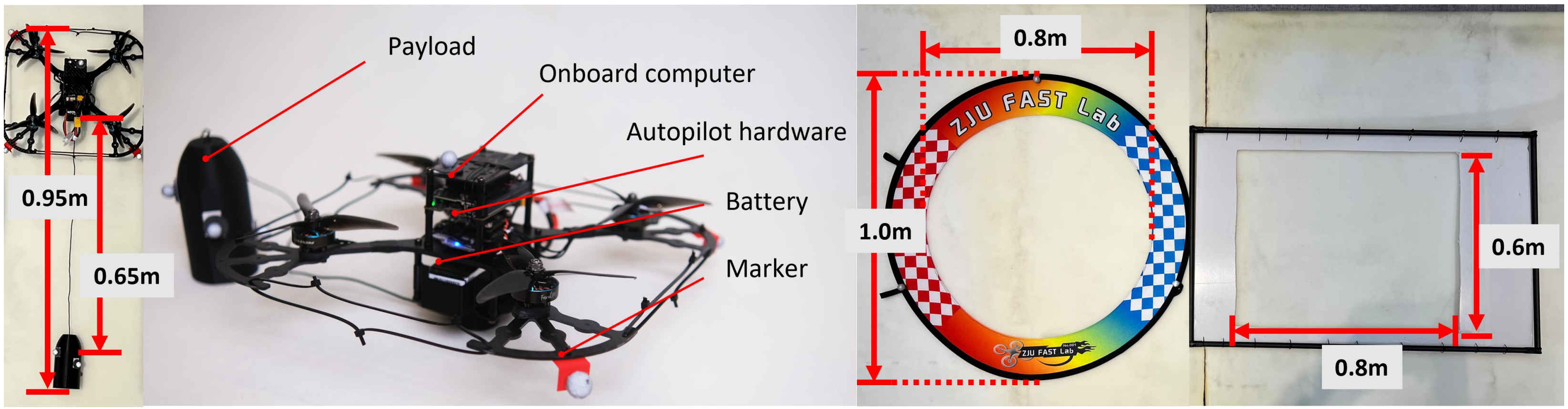
subject to

$$\boxed{\begin{aligned} \mathbf{g}(\mathbf{q}_{\mathbf{x}}^{[k]}(t)) &\leq \mathbf{0}, \quad (14b) \\ \mathbf{h}(\mathbf{q}_{\mathbf{x}}^{[k]}(t)) &= \mathbf{0}, \quad (14c) \end{aligned}}$$

$$\forall t \in [0, \|\mathbf{T}\|_1], \quad (14d)$$

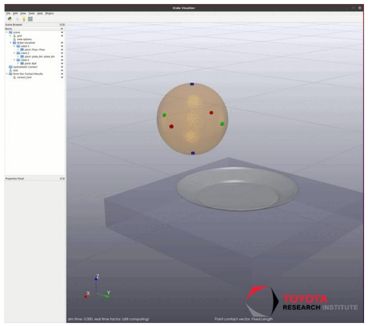
Type	Function	Parameters
Inequality	$g_o(t, \mathbf{c}) = \text{sign}_{\mathcal{E}}(\mathcal{M}_{r_{\text{safe}}}(t, \mathbf{c}))$	r_{safe}: the radius of safe margin.
Inequality	$g_v(t, \mathbf{c}) = \|\mathbf{v}(t, \mathbf{c})\|_2^2 - v_{\max}^2$	$v_{\max}$: the maximum velocity.
Inequality	$g_a(t, \mathbf{c}) = \|\mathbf{a}(t, \mathbf{c})\|_2^2 - a_{\max}^2$	$a_{\max}$: the maximum acceleration.
Inequality	$g_\theta(t, \mathbf{c}) = \theta - \theta_{\max}$	$\theta_{\max}$: the maximum angle of deviation.
Inequality	$g_u(t, \mathbf{c}) = \left(u(t, \mathbf{c}) - \frac{u_{\max} + u_{\min}}{2}\right)^2 - \left(\frac{u_{\max} - u_{\min}}{2}\right)^2$	$u_{\max}$: the maximum thrust, $u_{\min}$: the minimum thrust.
Inequality	$g_l(t, \mathbf{c}) = \left(l(t, \mathbf{c}) - \frac{l_0 + d_{\text{safe}}}{2}\right)^2 - \left(\frac{l_0 - d_{\text{safe}}}{2}\right)^2$	l_0: the length of cable, d_{safe}: the minimum safe distance.
Inequality	$g_{f_T}(t, \mathbf{c}) = \left(f_T(t, \mathbf{c}) - \frac{f_{T,\max}}{2}\right)^2 - \left(\frac{f_{T,\max}}{2}\right)^2$	$f_{T,\max}$: the maximum tension
Equality	$\mathbf{h}_{\text{dyn}}(t, \mathbf{c}) = \|\tilde{\mathbf{p}}(t, \mathbf{c})\| \cdot (\mathbf{a}_{\mathcal{L}}(t, \mathbf{c}) + g\mathbf{e}_z) + \|\mathbf{a}_{\mathcal{L}}(t, \mathbf{c}) + g\mathbf{e}_z\| \cdot \tilde{\mathbf{p}}(t, \mathbf{c})$	$\tilde{\mathbf{p}}(t, \mathbf{c})$: the vector pointing from the quadrotor and the payload. g: the gravity, $\mathbf{e}_z$: the z-axis direction of the world frame.
Equality	$h_{\text{comp}}(t, \mathbf{c}) = T(t, \mathbf{c}) \cdot (l_0 - l(t, \mathbf{c}))$	l_0: the length of cable

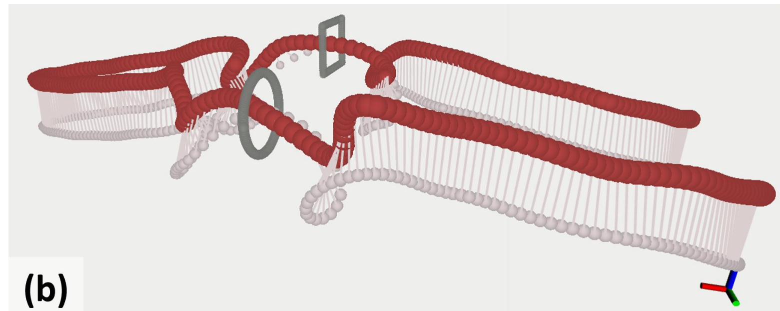
System Setup

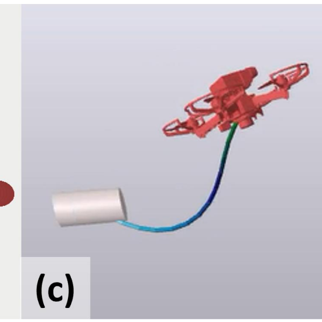


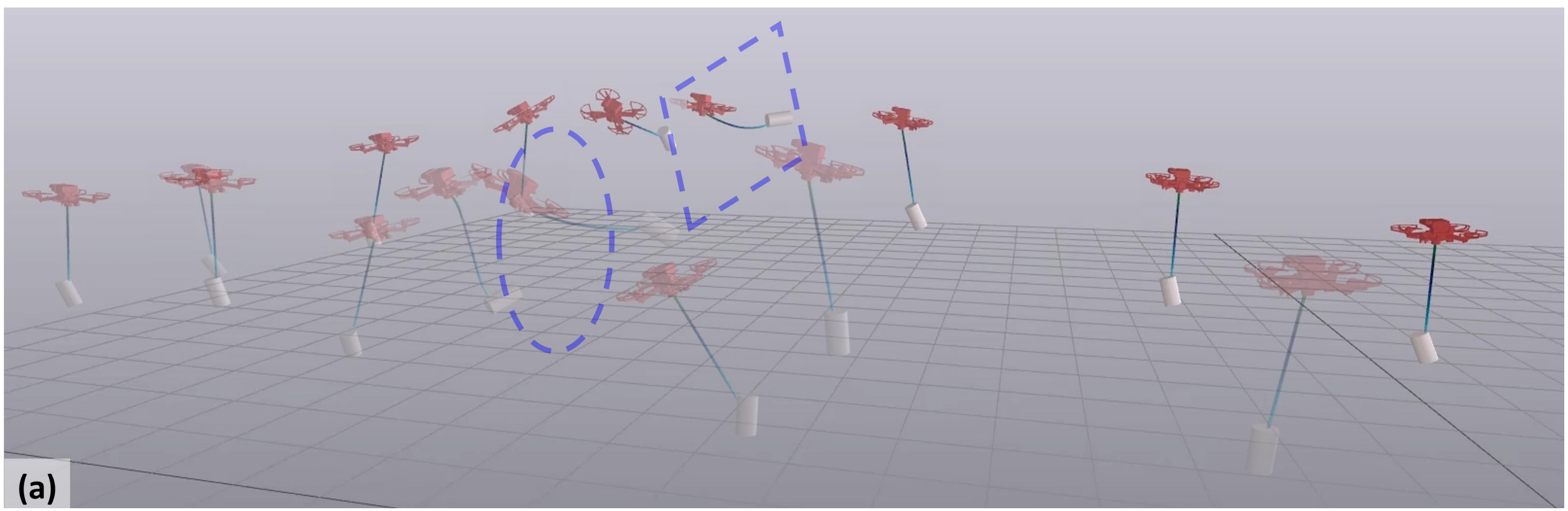
Notation	Value	Description
m_Q	0.746 kg	Mass of the quadrotor
$m_\mathcal{L}$	0.054 kg	Mass of the payload
l_0	0.644 m	Length of cable
r_Q	0.2 m	Safe margin radius of the quadrotor
$r_\mathcal{L}$	0.2 m	Safe margin radius of the payload

Demonstrations

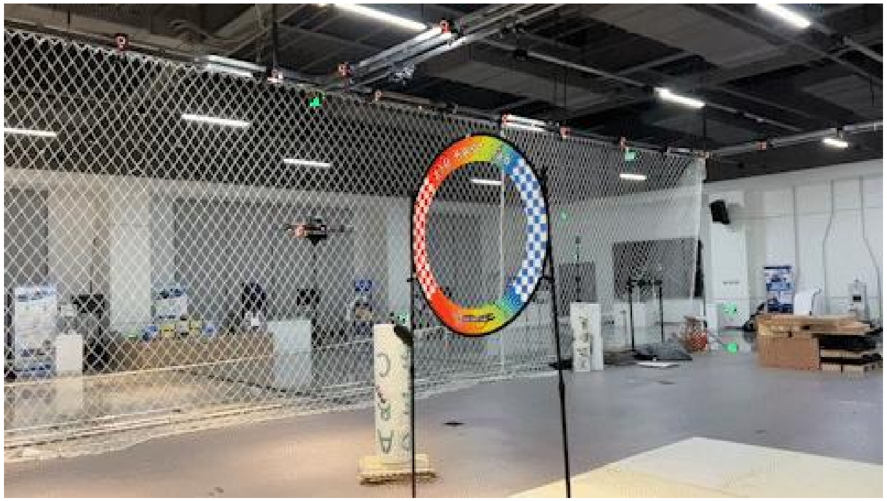


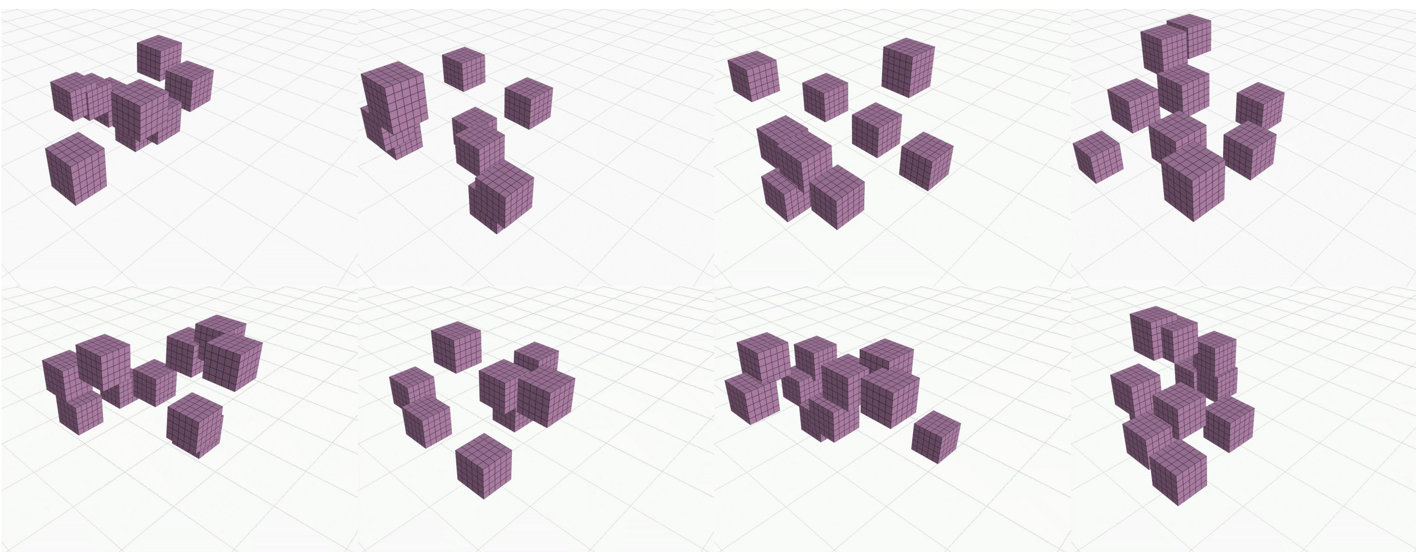

(b)

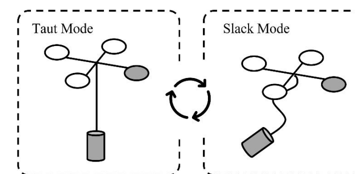

(c)


(a)

Demonstrations

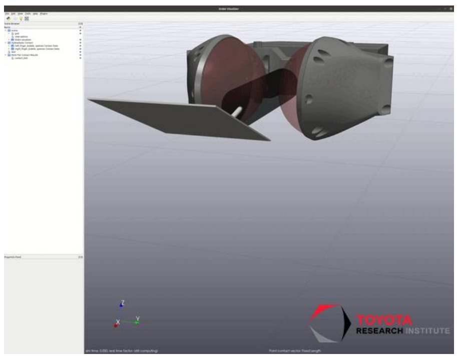


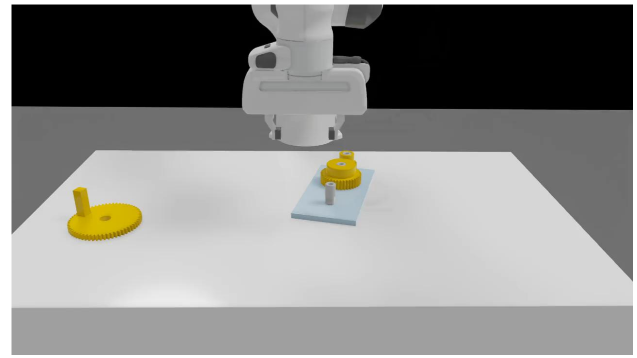


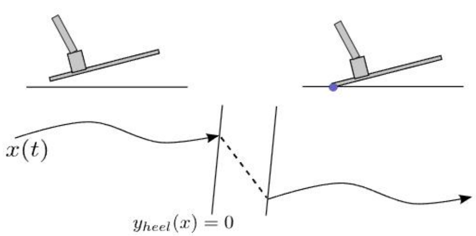


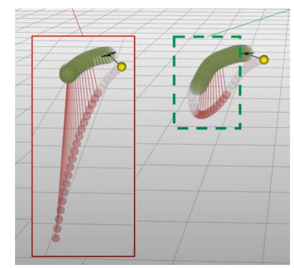
Complex Scenarios

Motivation: Demand for Contact-Aware Motion Planning









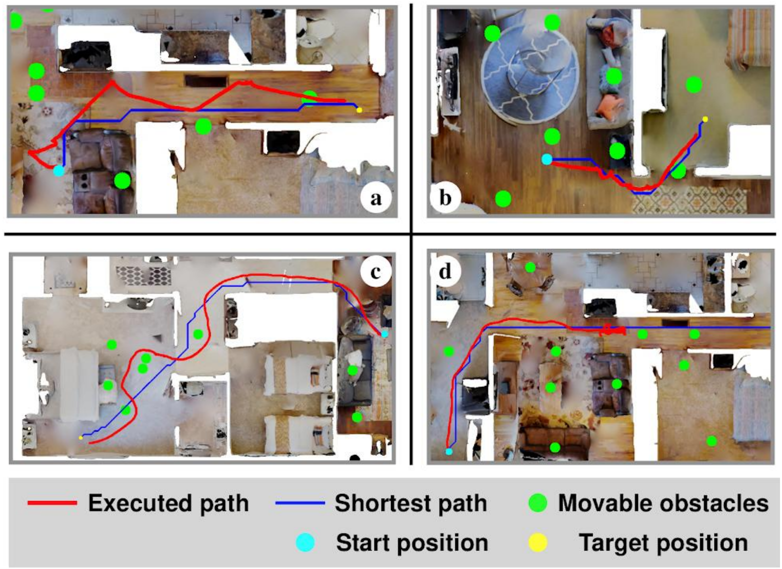
Modes Switching

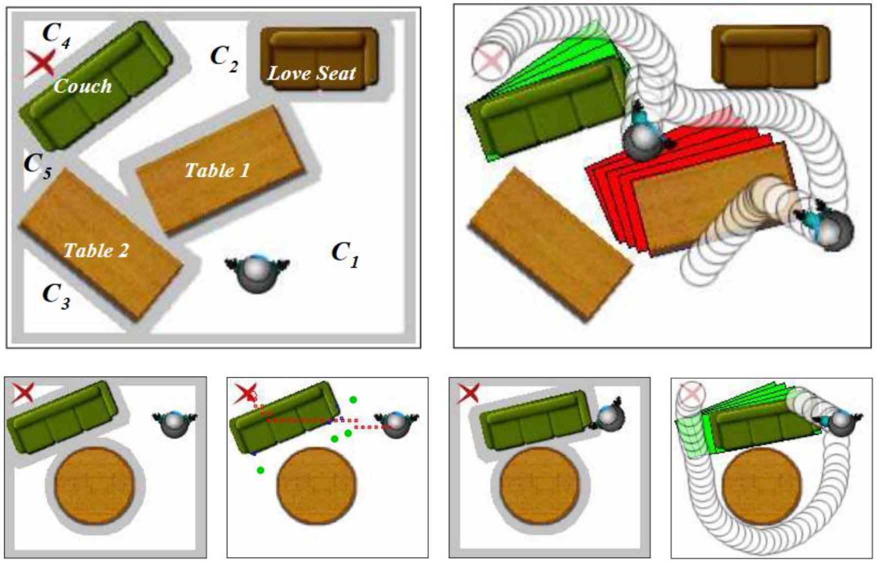
Make / Brake Collision

[1] M. Sherman, “Rethinking contact simulation for robot manipulation,” Medium, https://medium.com/toyotaresearch/rethinking-contact-simulation-for-robot-manipulation-434a56b5ec88 (accessed Oct. 27, 2023).
[2] Russ Tedrake. Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832).

NAMO

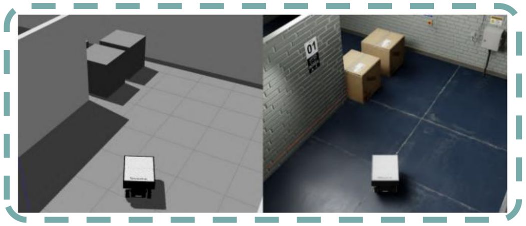
Navigation **A**mong **M**ovable **O**bjects (NAMO)[1,2]:





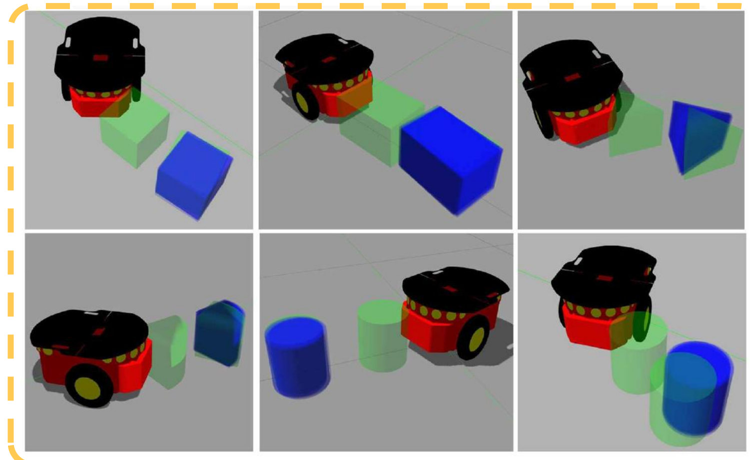
[1] B. Shen et al., "iGibson 1.0: A Simulation Environment for Interactive Tasks in Large Realistic Scenes," 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 2021, pp. 7520-7527, doi: 10.1109/IROS51168.2021.9636667.
[2] M. Stilman, and J. Kuffner. "Planning Among Movable Obstacles with Artificial Constraints," The International Journal of Robotics Research. 2008, 27(11-12):1295-1307. doi:10.1177/0278364908098457

Motion Planning Among Movable Objects



NAMO

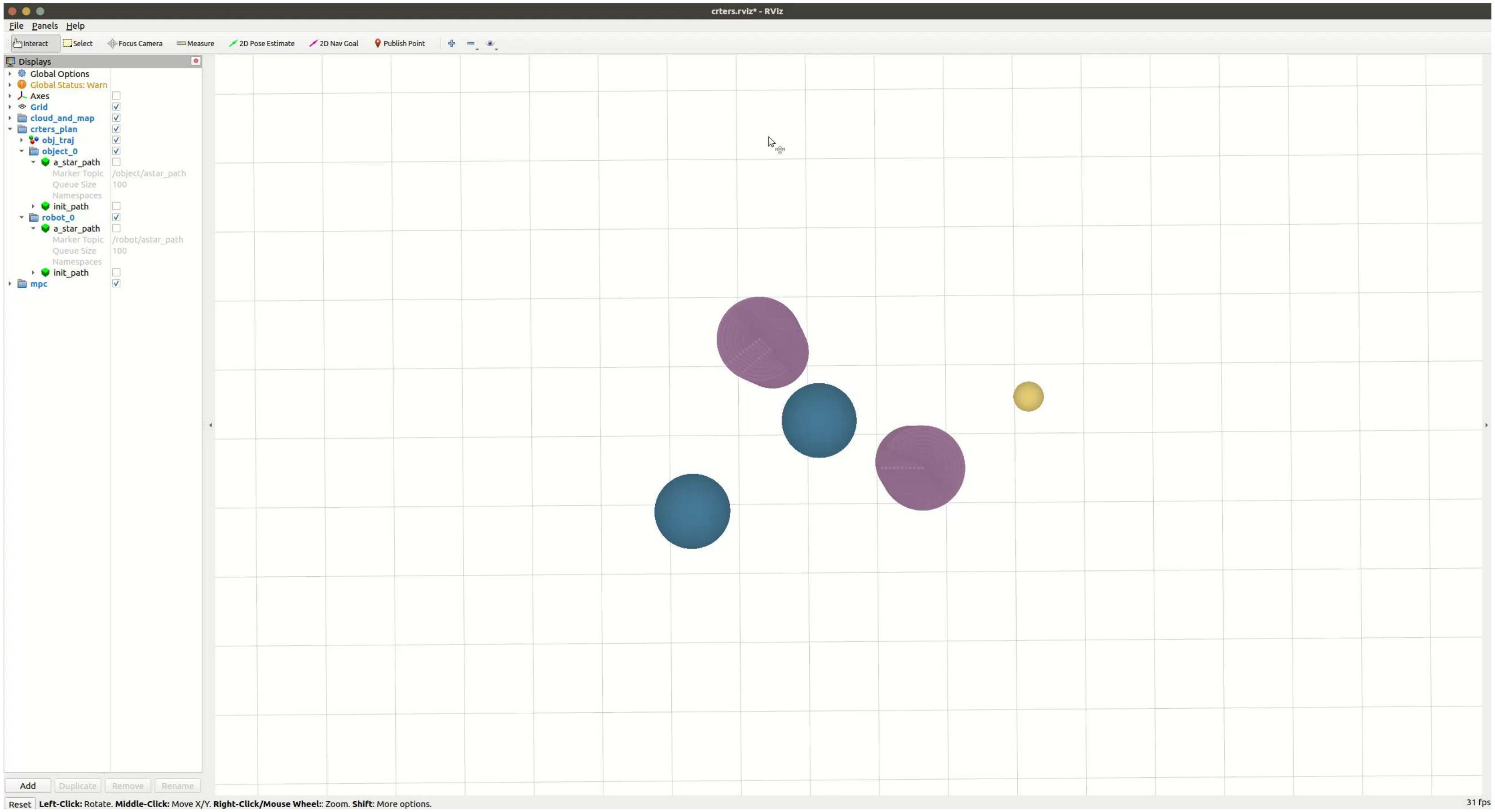
RAMO



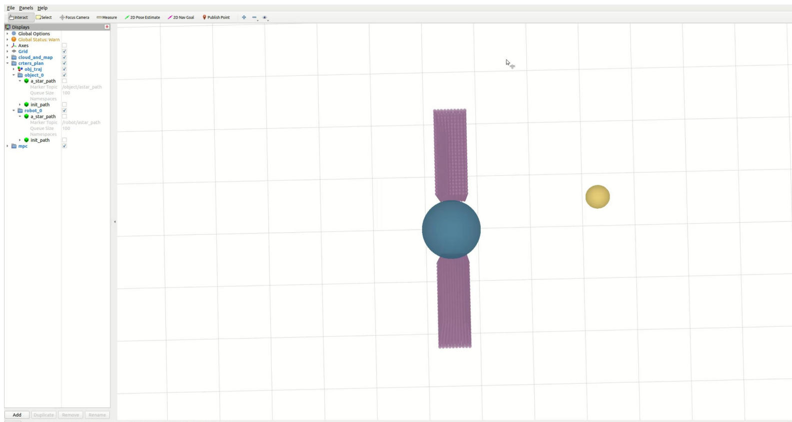
1. Determining ***which objects*** and ***where*** they need to be moved is difficult.
   - Because the size of the configuration space, which consists of the robot in conjunction with the objects, increases exponentially with the number of objects.
2. Modeling the ***system dynamics*** of the robot will become very complex because of the ***interactions*** between the robot and the environment.
   - For example, we need to consider the dynamics model of the robot in at least two cases, where contact occurs and where no contact occurs.
3. How to efficiently solve such a complex optimization problem remains an open question.

[1] J. St¨uber, C. Zito, and R. Stolkin, "Let's push things forward: A survey on robot pushing," Frontiers in Robotics and AI, vol. 7, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2020.00008

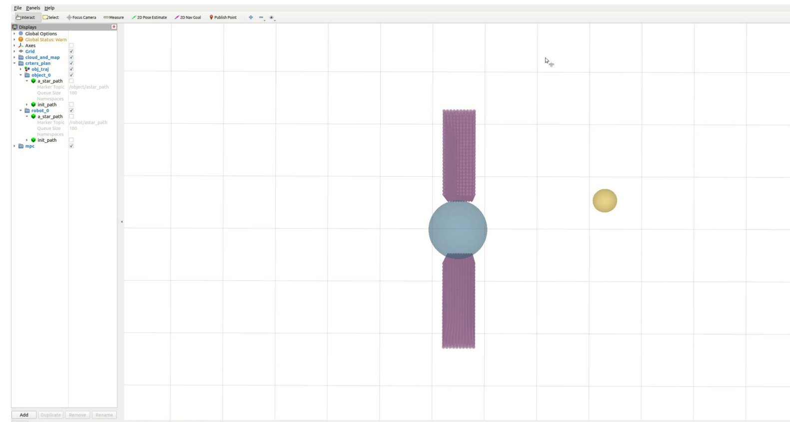
Motion Planning Among Movable Objects



Motion Planning Among Movable Objects



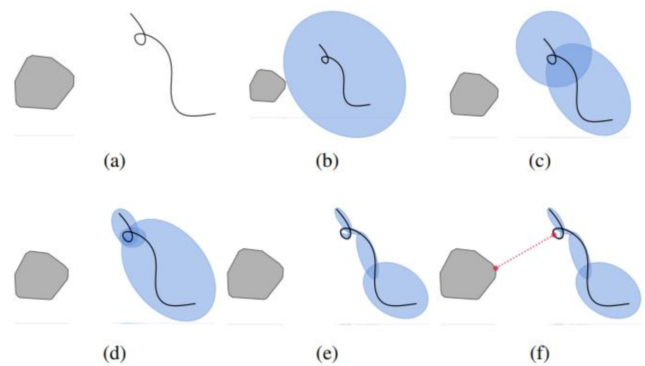
w/o $\mathcal{K}(\mathbf{o}, \mathbf{q}, \mathbf{T})$



with $\mathcal{K}(\mathbf{o}, \mathbf{q}, \mathbf{T})$

$$\mathbf{x} = [\underbrace{o_0, \dots, o_{N-1}}_{\mathbf{o}}, \underbrace{q_{\mathrm{r},0}, \dots, q_{\mathrm{r},M-1}, q_{\mathrm{o}_0,0}, \dots}_{\mathbf{q}}, \underbrace{\tau_0, \dots, \tau_{M-1}}_{\mathbf{t}}]^T$$

$$\mathbf{x}^* = \arg\min_{\mathbf{o},\mathbf{q},\mathbf{T}} \mathcal{F}(\mathbf{q}, \mathbf{T}) + \boxed{\mathcal{K}(\mathbf{o}, \mathbf{q}, \mathbf{T})} + \mathcal{L}_{\mu,\lambda,\rho}(\mathbf{o}, \mathbf{q}, \mathbf{T})$$



Q & A

Haokun WANG
E-mail: hwangeh@connect.ust.hk