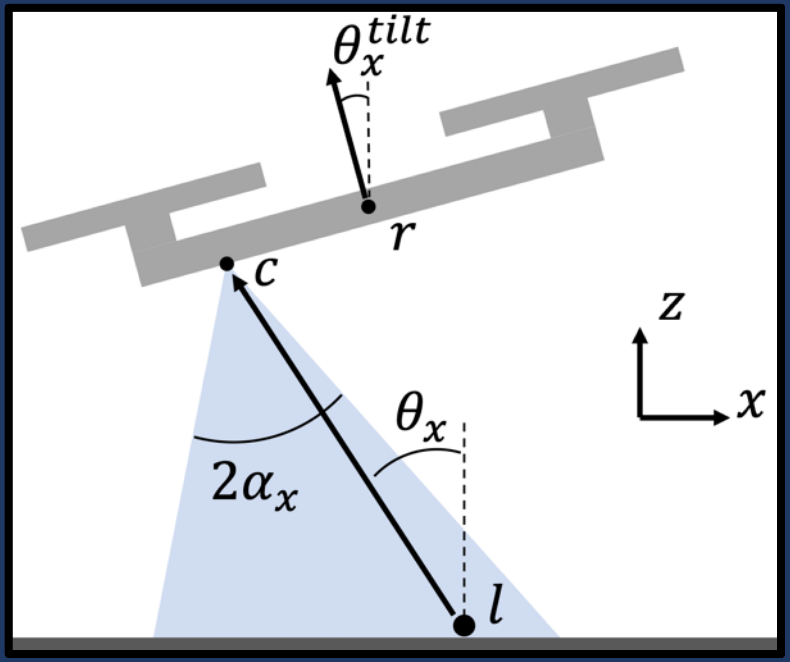
Vision-based Autonomous Landing of a Quadcopter with Field-of-View Constraints



Sequoyah Walters

8/14/23

ARC Lab
Autonomous & Resilient Controls Lab

Motivation

- Drone delivery services, search-and-rescue operations, collaborative robotics.
- Inadequate visibility of landing pad in quadcopter-mounted camera causes **landing inaccuracy**, **collisions**, and a **decrease in safety**.

This thesis: Planning and control that ensures landing pad visibility with a quadcopter-mounted camera

Overview

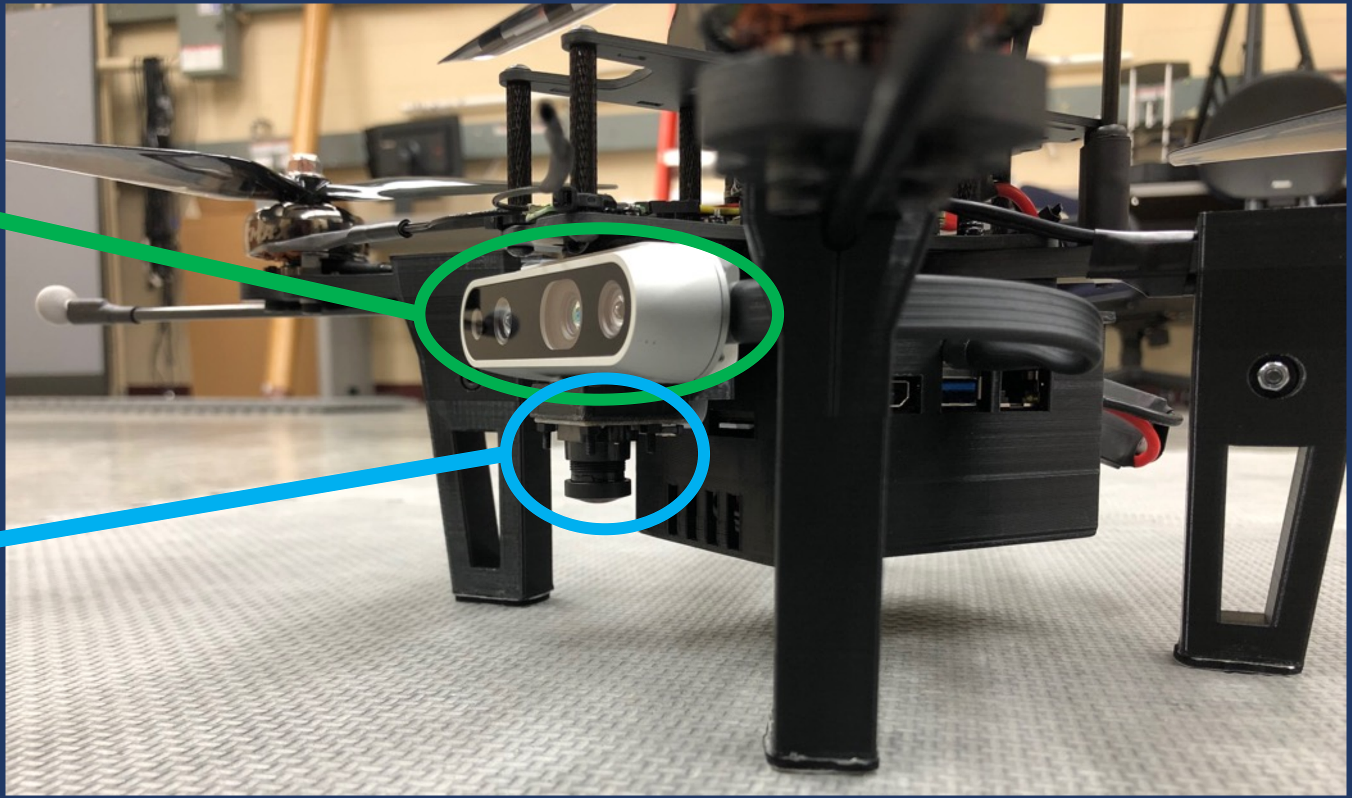
Hardware

Stereo camera with inertial measurement unit (IMU)

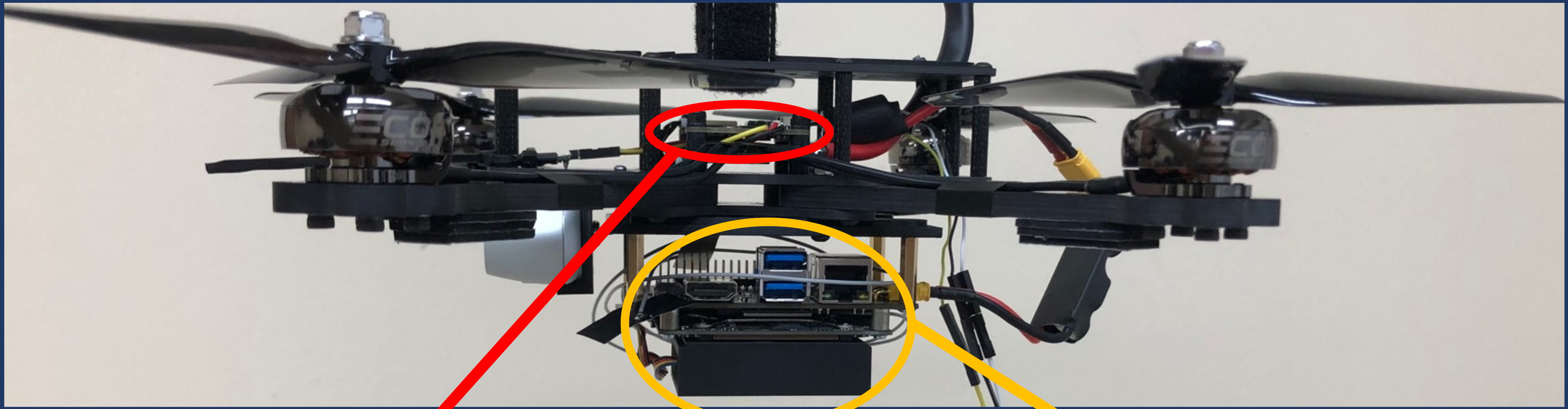
- used for visual-inertial odometry (VIO)

Down-facing **monocular** camera

- used for detecting landing pad



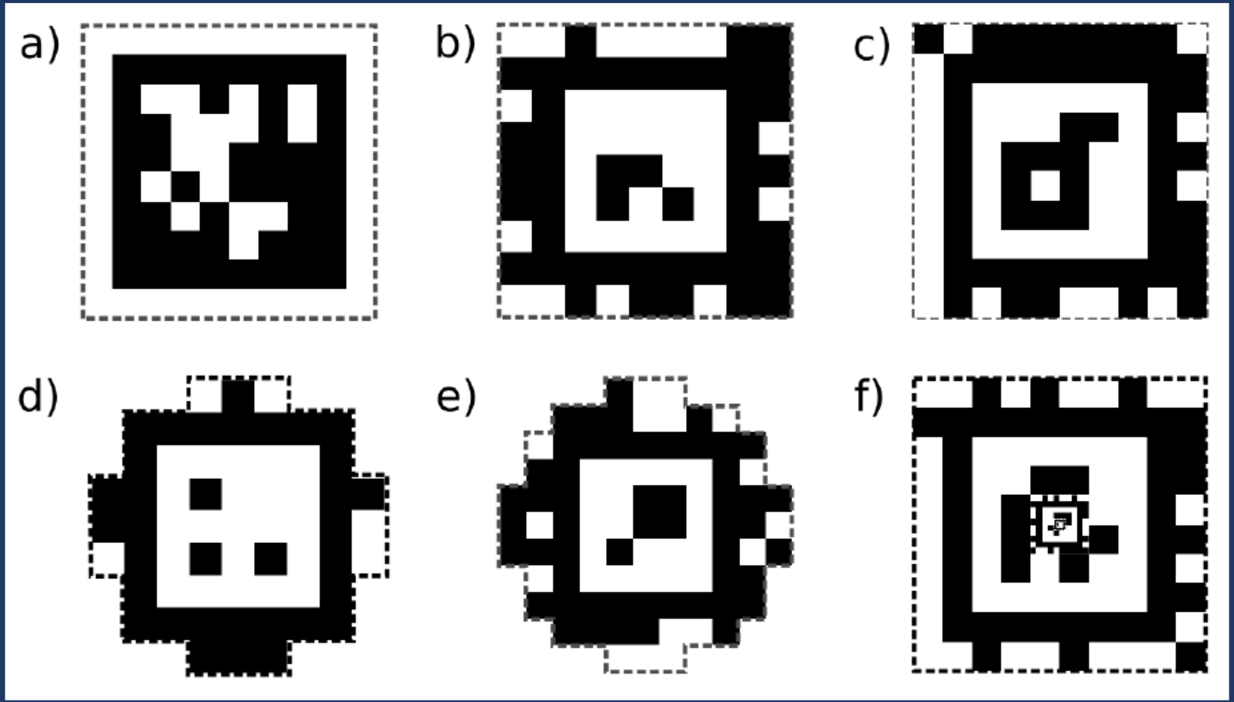
Hardware



Flight control unit (FCU) ⇄ Companion computer (CC)

UART

AprilTag landing pad



The AprilTag algorithm [1] is used to efficiently and accurately detect AprilTag patterns

AprilTag marker used as landing pad

[1]: Krogius et al. “Flexible Layouts for Fiducial Tags”. 2019

Landing

1. State Estimation:
   - Visual-inertial odometry (VIO) used to estimate quadcopter position, orientation and velocity (up to ~100 Hz using IMU post-integration).

Landing

1. State Estimation:
   - Visual-inertial odometry (VIO) used to estimate quadcopter position, orientation and velocity (up to ~100 Hz using IMU post-integration).
2. Landing pad detection:
   - AprilTag algorithm detects the AprilTag marker using down-facing camera (~20 Hz).

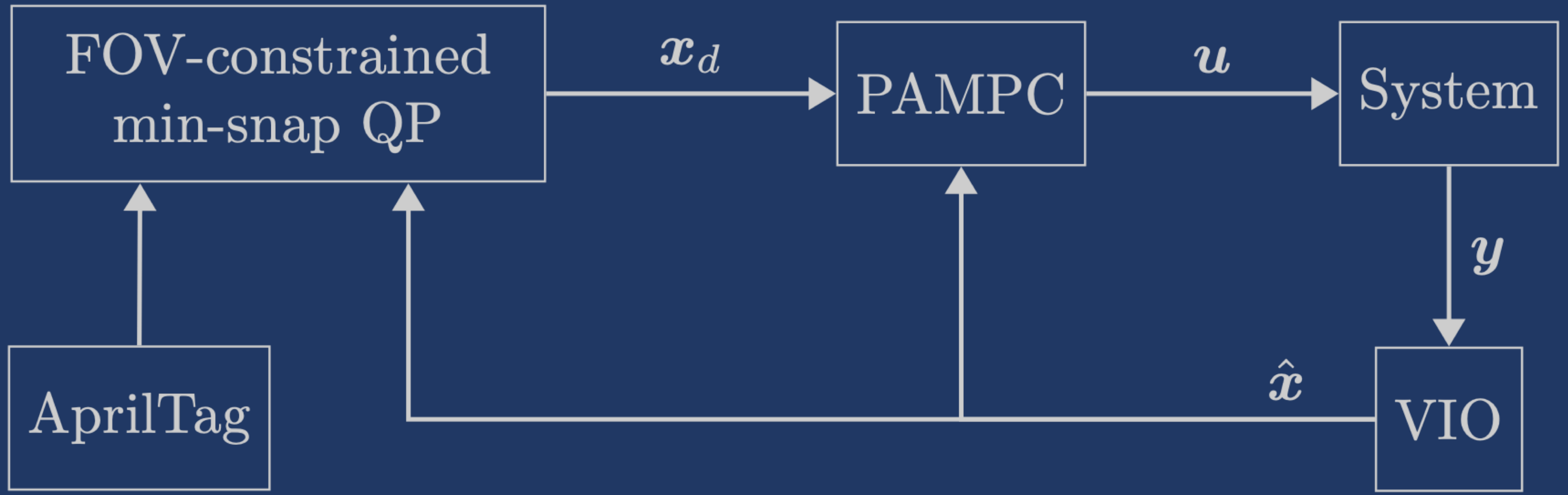
Landing

1. State Estimation:
   - Visual-inertial odometry (VIO) used to estimate quadcopter position, orientation and velocity (up to ~100 Hz using IMU post-integration).
2. Landing pad detection:
   - AprilTag algorithm detects the AprilTag marker using down-facing camera (~20 Hz).
3. Trajectory generation:
   - A minimum-snap quadratic program (QP) generates a FOV-constrained landing trajectory (~30ms for each generation).

Landing

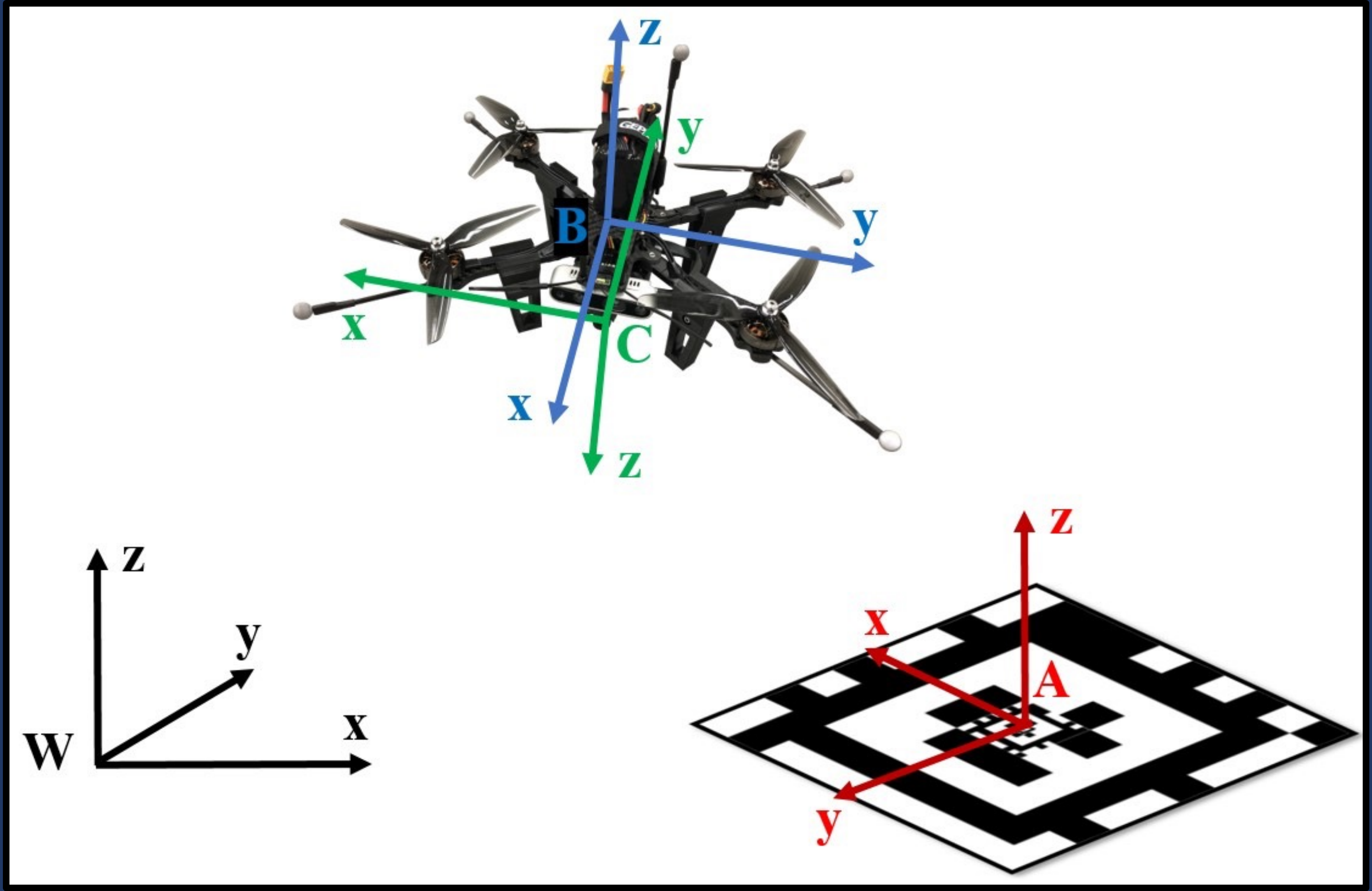
1. State Estimation:
   - Visual-inertial odometry (VIO) used to estimate quadcopter position, orientation and velocity (up to ~100 Hz using IMU post-integration).
2. Landing pad detection:
   - AprilTag algorithm detects the AprilTag marker using down-facing camera (~20 Hz).
3. Trajectory generation:
   - A minimum-snap quadratic program (QP) generates a FOV-constrained landing trajectory (~30ms for each generation).
4. Tracking control:
   - Perception-aware model-predictive control (PAMPC) method used to track the landing trajectory (fixed at 50Hz).

Landing


FOV-constrained
min-snap QP
x_d
PAMPC
u
System
y
AprilTag
$\hat{x}$
VIO

Background

Coordinate Frames



Quadcopter dynamics (from [2])

- State:

$$\boldsymbol{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & q_w & q_x & q_y & q_z \end{bmatrix}^T \in \mathbb{R}^{10}$$
$$= \begin{bmatrix} \boldsymbol{r}^T & \boldsymbol{v}^T & \boldsymbol{q}^T \end{bmatrix}^T,$$

position — velocity — orientation

[2]: Falanga et al. “PAMPC: Perception-Aware Model Predictive Control for Quadrotors”. 2018

Quadcopter dynamics (from [2])

- State:

$$\boldsymbol{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & q_w & q_x & q_y & q_z \end{bmatrix}^T \in \mathbb{R}^{10}$$
$$= \begin{bmatrix} \boldsymbol{r}^T & \boldsymbol{v}^T & \boldsymbol{q}^T \end{bmatrix}^T,$$

- Control inputs:

$$\boldsymbol{u} = \begin{bmatrix} \tau & \boldsymbol{\omega}^T \end{bmatrix}^T \in \mathbb{R}^4.$$

mass-normalized thrust   angular velocity

[2]: Falanga et al. "PAMPC: Perception-Aware Model Predictive Control for Quadrotors". 2018

Quadcopter dynamics (from [2])

- System dynamics:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) = \begin{bmatrix} \dot{\boldsymbol{r}} \\ \dot{\boldsymbol{v}} \\ \dot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{g} + Rot(\boldsymbol{q})\boldsymbol{\tau} \\ \frac{1}{2}\Lambda(\boldsymbol{\omega})\boldsymbol{q} \end{bmatrix}, \qquad (1)$$

- Gravity expressed in the world frame: $\boldsymbol{g} := [0 \;\; 0 \; -g]$
- Thrust expressed in the body frame: $\boldsymbol{\tau} := [0 \;\; 0 \;\; \tau]^T$

[2]: Falanga et al. “PAMPC: Perception-Aware Model Predictive Control for Quadrotors”. 2018

Quadcopter dynamics (from [2])

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{g} + Rot(\boldsymbol{q})\boldsymbol{\tau} \\ \frac{1}{2}\Lambda(\boldsymbol{\omega})\boldsymbol{q} \end{bmatrix}$$

- Quaternion to rotation matrix operator:

$$Rot(\boldsymbol{q}) := \begin{bmatrix} 1-2q_y^2-2q_z^2 & 2\left(q_xq_y+q_wq_z\right) & 2\left(q_xq_z-q_wq_y\right) \\ 2\left(q_xq_y-q_wq_z\right) & 1-2q_x^2-2q_z^2 & 2\left(q_yq_z+q_wq_x\right) \\ 2\left(q_xq_z+q_wq_y\right) & 2\left(q_yq_z-q_wq_x\right) & 1-2q_x^2-2q_y^2 \end{bmatrix},$$

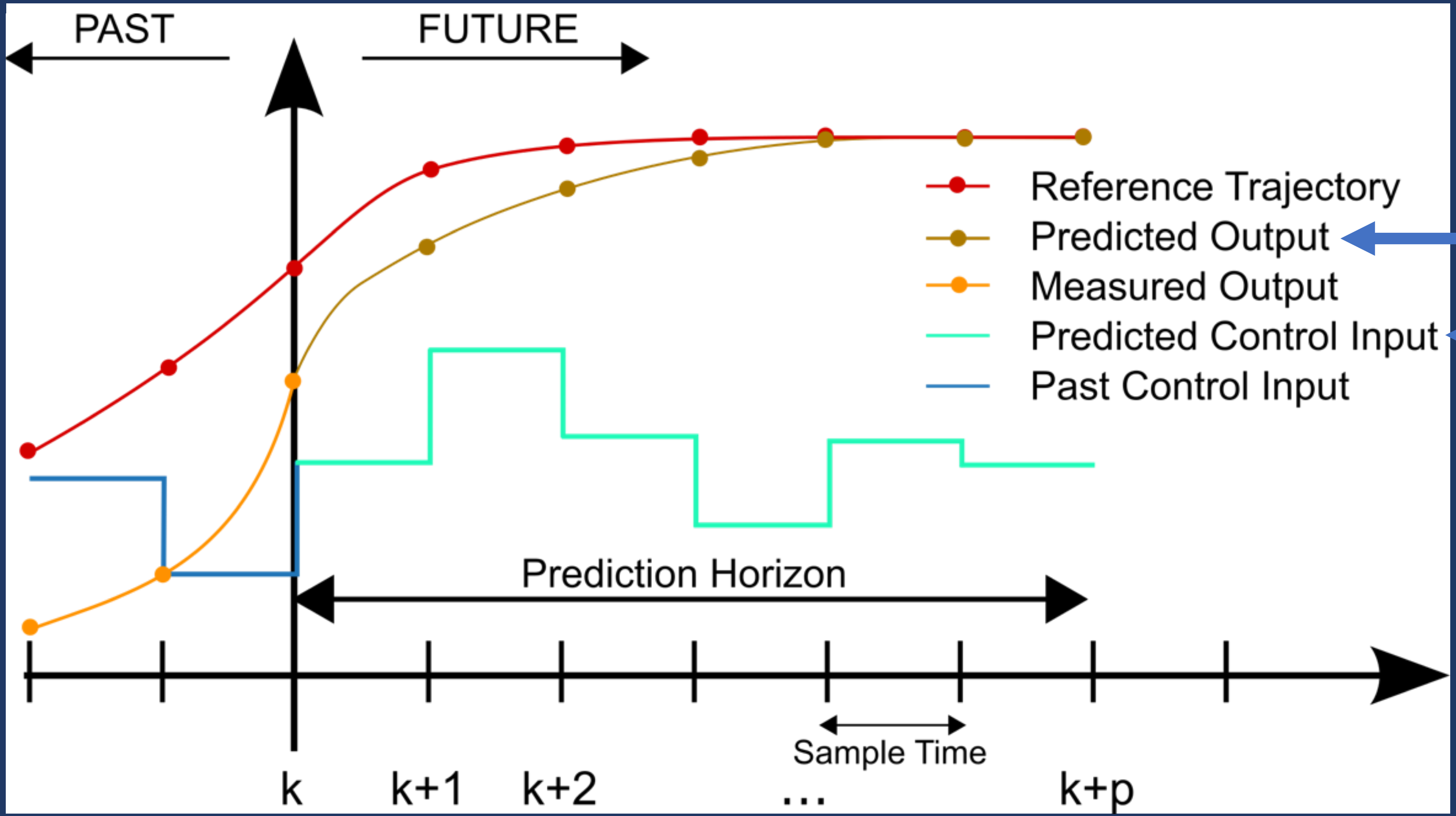
- Skew-symmetric matrix:

$$\Lambda(\boldsymbol{\omega}) := \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}.$$

[2]: Falanga et al. “PAMPC: Perception-Aware Model Predictive Control for Quadrotors”. 2018

Model-predictive control (MPC)

The idea: **iterative** control method where model is used to **predict** the future behavior of a system over a **finite time window**. Can be used to track a reference trajectory:



Solve optimization problem to find these

Model-predictive control (MPC)

Some setup for the optimization problem:

Discretization of dynamics (1):

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t))$$

$$\downarrow$$

$$\boldsymbol{x}(t_{i+1}) = F(\boldsymbol{x}(t_i), \boldsymbol{u}(t_i))$$

Discretization time step: $\Delta t := t_i - t_{i-1},\ \forall i \in \{1, \dots, N\}$

Model-predictive control (MPC)

$$\boldsymbol{z}_N := \left[\boldsymbol{x}(t_N) - \boldsymbol{x}_d(t_N)\right] \text{ and } \boldsymbol{z}(t) := \begin{bmatrix} \boldsymbol{x}(t) - \boldsymbol{x}_d(t) \\ \boldsymbol{u}(t) \end{bmatrix}$$

$$\min_{\substack{\boldsymbol{x}(t_1), \ldots, \boldsymbol{x}(t_N) \\ \boldsymbol{u}(t_0), \ldots, \boldsymbol{u}(t_{N-1})}} \quad \boldsymbol{z}_N^T Q_N \boldsymbol{z}_N + \sum_{i=0}^{N-1} \boldsymbol{z}(t_i)^T Q \boldsymbol{z}(t_i) \tag{2.1}$$

weight matrices

$$\text{s.t.} \quad \boldsymbol{x}(t_0) = \hat{\boldsymbol{x}}(t_0) \tag{2.2}$$

$$\boldsymbol{x}(t_{i+1}) = F(\boldsymbol{x}(t_i), \boldsymbol{u}(t_i)), \quad \forall i \in \{0, \ldots, N-1\} \tag{2.3}$$

$$\boldsymbol{x}(t_i) \in \mathcal{X}, \quad \forall i \in \{0, \ldots, N\} \tag{2.4}$$

$$\boldsymbol{u}(t_i) \in \mathcal{U}, \quad \forall i \in \{0, \ldots, N-1\} \tag{2.5}$$

state constraint

control input constraint

Minimum-snap trajectory generation

The idea: optimize over the coefficients of a piecewise polynomial to efficiently generate a trajectory for a quadcopter system [3].

- Piecewise polynomial used here:

$$\boldsymbol{\sigma}_T(t) = \begin{bmatrix} \mathbf{r}_T(t) \\ \psi_T(t) \end{bmatrix} = \begin{bmatrix} x_T(t) \\ y_T(t) \\ z_T(t) \\ \psi_T(t) \end{bmatrix} := \begin{cases} \sum_{i=0}^{n} \boldsymbol{\sigma}_{Ti1} t^i & t_0 \leq t < t_1 \\ \sum_{i=0}^{n} \boldsymbol{\sigma}_{Ti2} t^i & t_1 \leq t < t_2 \\ \vdots & \\ \sum_{i=0}^{n} \boldsymbol{\sigma}_{Tim} t^i & t_{m-1} \leq t \leq t_m \end{cases},$$

quadcopter trajectory

position and yaw

coefficients

[3]: Mellinger et al. “Minimum snap trajectory generation and control for quadrotors”. 2011

$$\min_{\substack{\boldsymbol{r}_{T_{ij}},\ \psi_{T_{ij}} \\ \forall i \in \{0,\ldots,n\} \\ \forall j \in \{1,\ldots,m\}}} \int_{t_0}^{t_m} \mu_r \left\| \mathbf{r}_T^{(k_r)}(t) \right\|^2 + \mu_\psi \left(\psi_T^{(k_\psi)}(t) \right)^2 dt$$

Min-snap objective (when $k_r = 4$)

$$\text{s.t.} \quad \boldsymbol{\sigma}_T(t_j) = \boldsymbol{\sigma}_j, \quad j = 0, \ldots, m$$

waypoints

boundary conditions

$$\mathbf{r}_T^{(p)}(t_j) = \mathbf{r}_j^{(p)} \text{ or free}, \quad j = \{0, m\};\ p = 1, \ldots, k_r$$

$$\psi_T^{(p)}(t_j) = \psi_j^{(p)} \text{ or free}, \quad j = \{0, m\};\ p = 1, \ldots, k_\psi$$

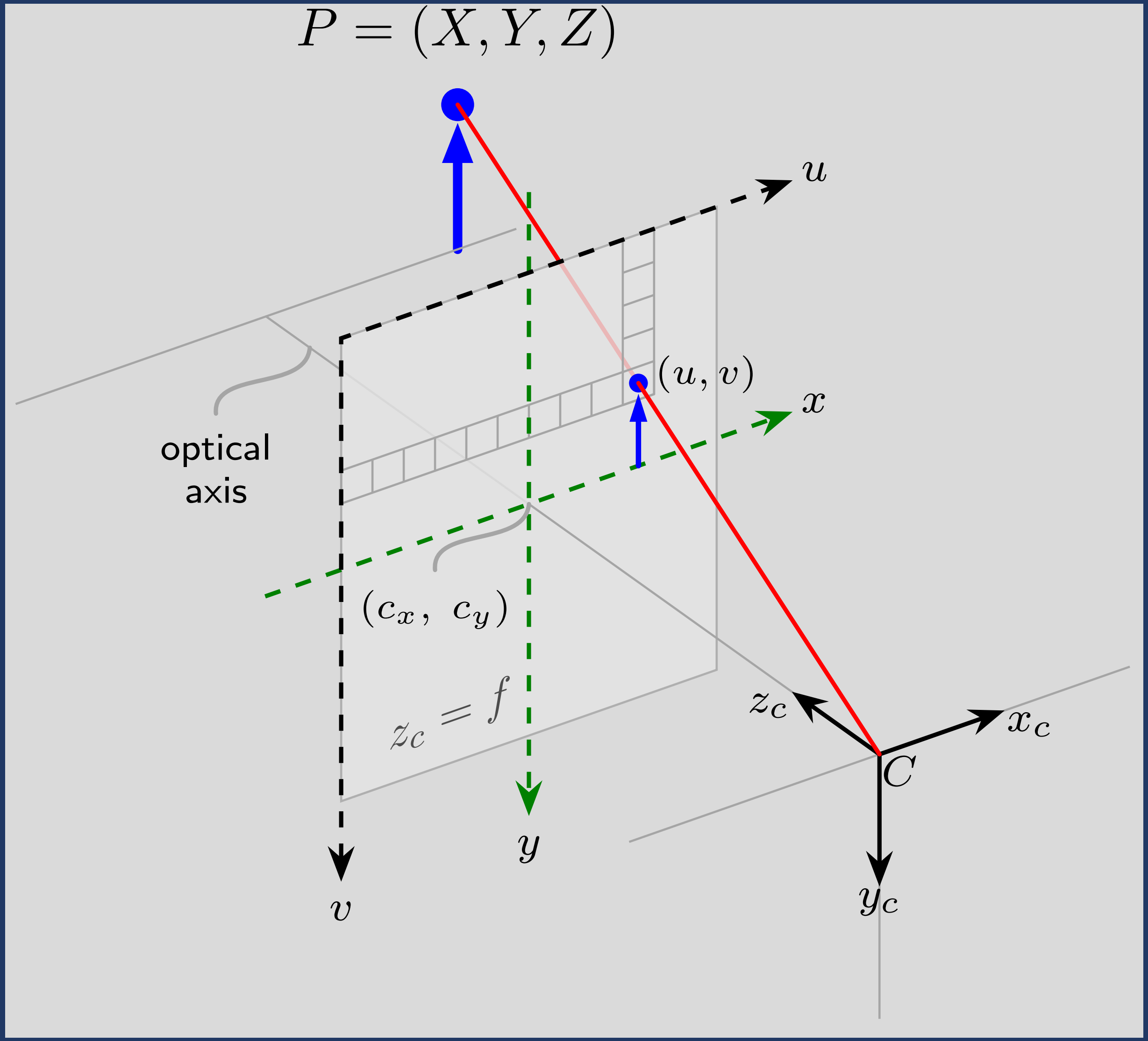
continuity

$$\sum_{i=p}^{n} (\boldsymbol{r}_{T_{ij}} - \boldsymbol{r}_{T_{i,j+1}}) = 0, \quad j = 1, \ldots, m-1;\ p = 0, \ldots, k_r$$

$$\sum_{i=p}^{n} (\psi_{T_{ij}} - \psi_{T_{i,j+1}}) = 0, \quad j = 1, \ldots, m-1;\ p = 0, \ldots, k_\psi$$

[3]: Mellinger et al. “Minimum snap trajectory generation and control for quadrotors”. 2011

Pinhole camera model (approximation of a perspective camera)



- The pinhole model is used for VO/VIO and the perception-aware MPC method

$$u = f_x \frac{X}{Z} + c_x$$

$$v = f_y \frac{Y}{Z} + c_y$$

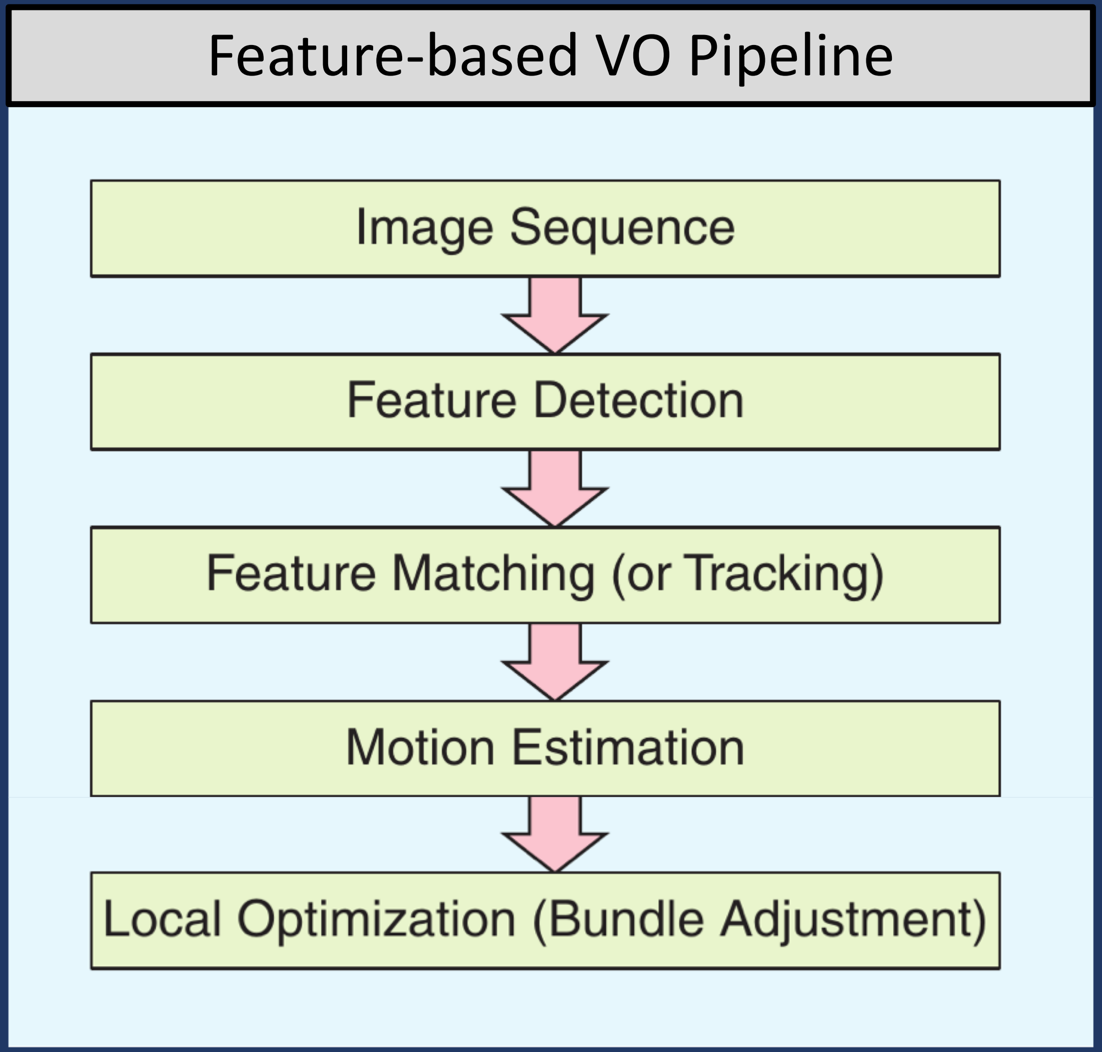
f_x : horizontal focal length (pixels)

f_y : vertical focal length (pixels)

Visual odometry (VO)

The idea: use one or more cameras for pose (position and orientation) estimation.

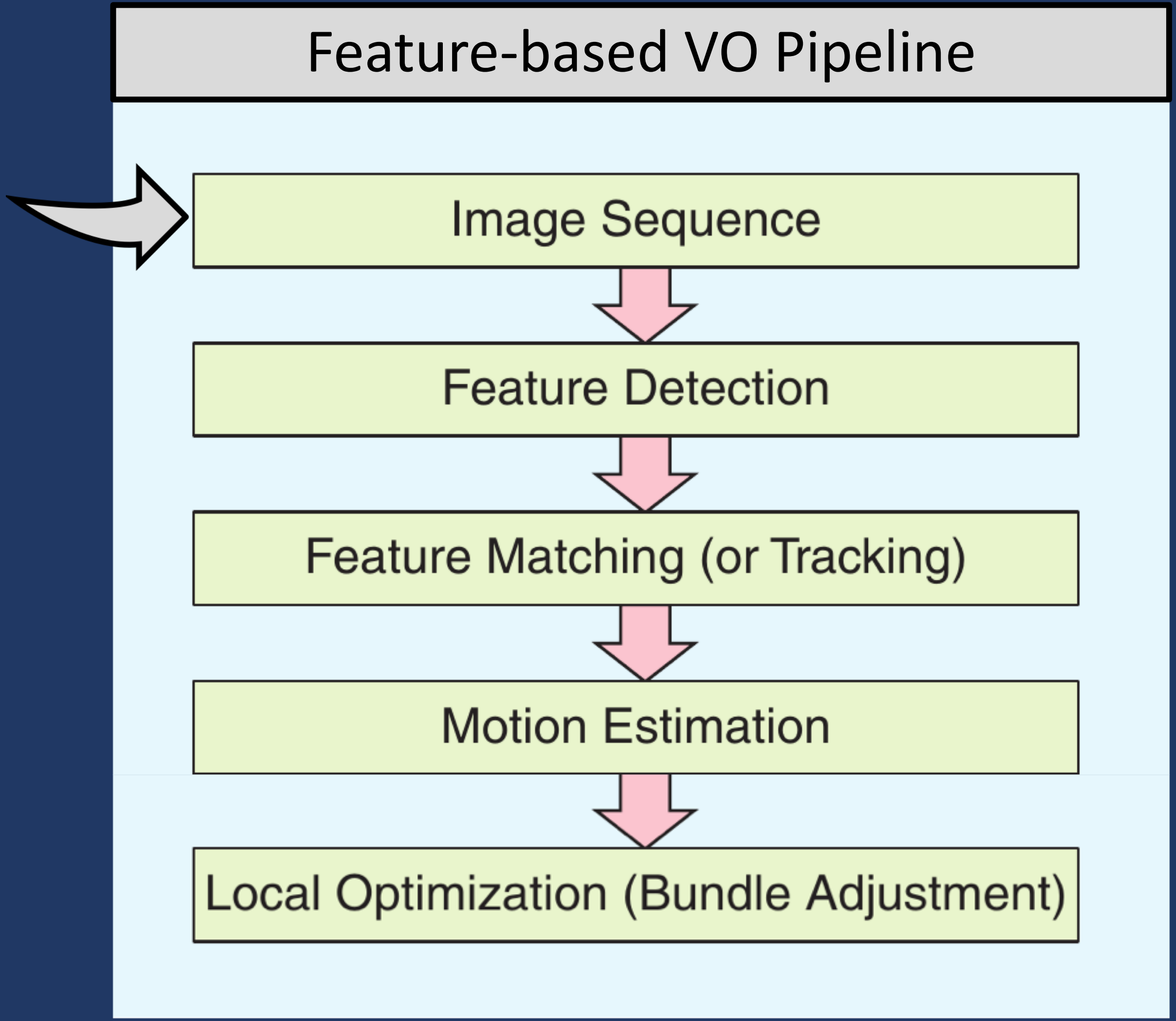
- No IMU is used.
- Helpful to understand VO before VIO



I_k

I_{k+1}

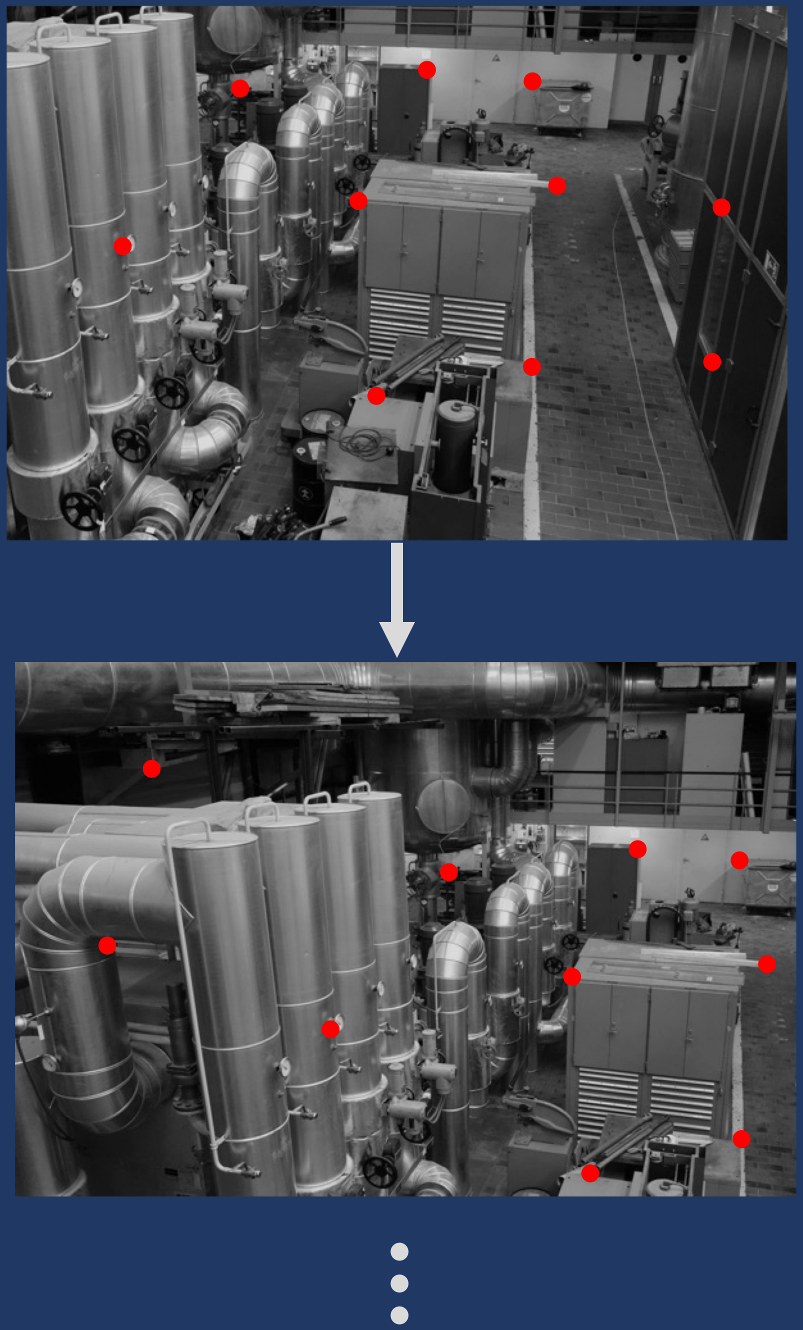
⋮

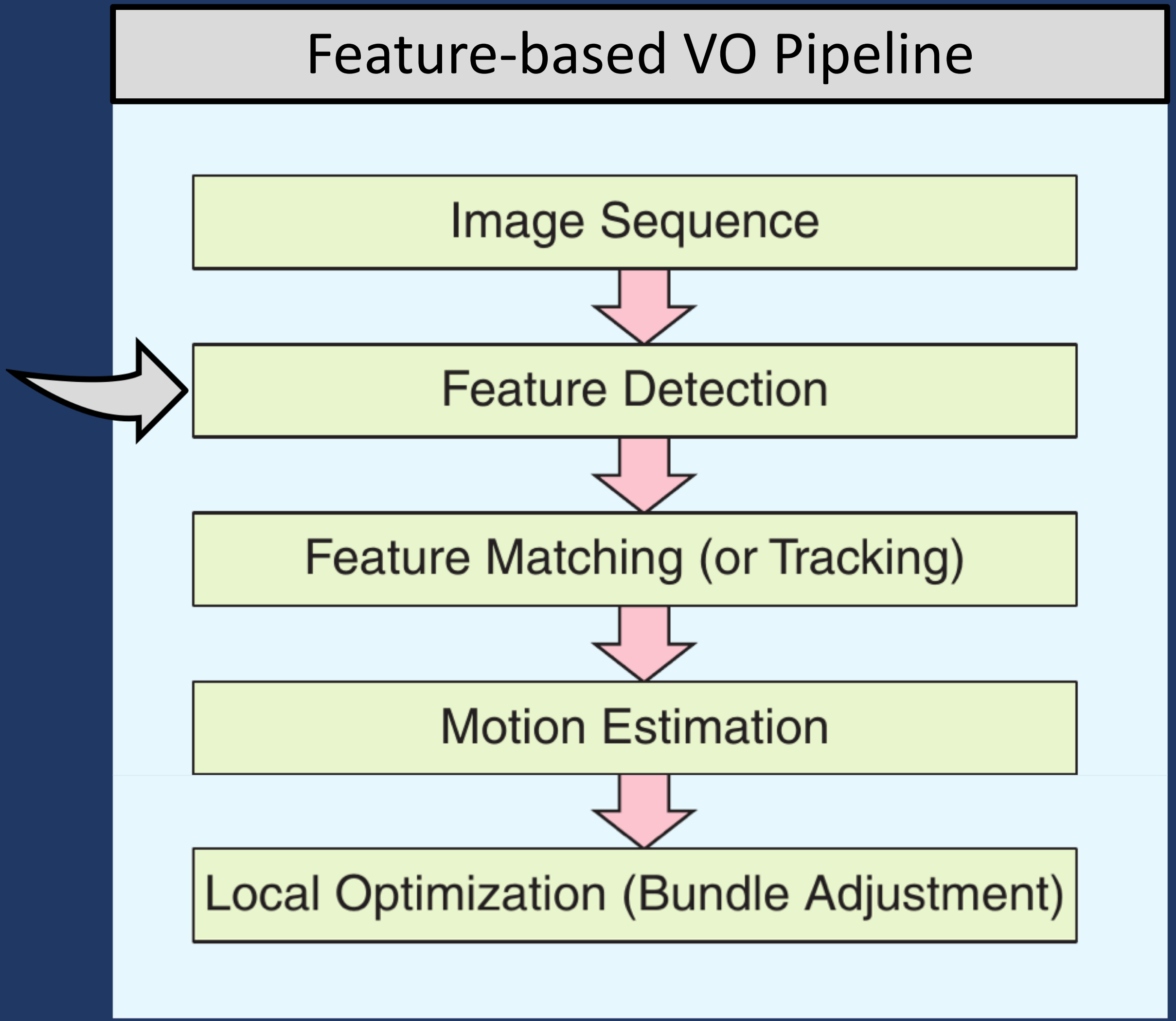

Feature-based VO Pipeline
Image Sequence
Feature Detection
Feature Matching (or Tracking)
Motion Estimation
Local Optimization (Bundle Adjustment)

ARC Lab
Autonomous & Resilient Controls Lab

I_k

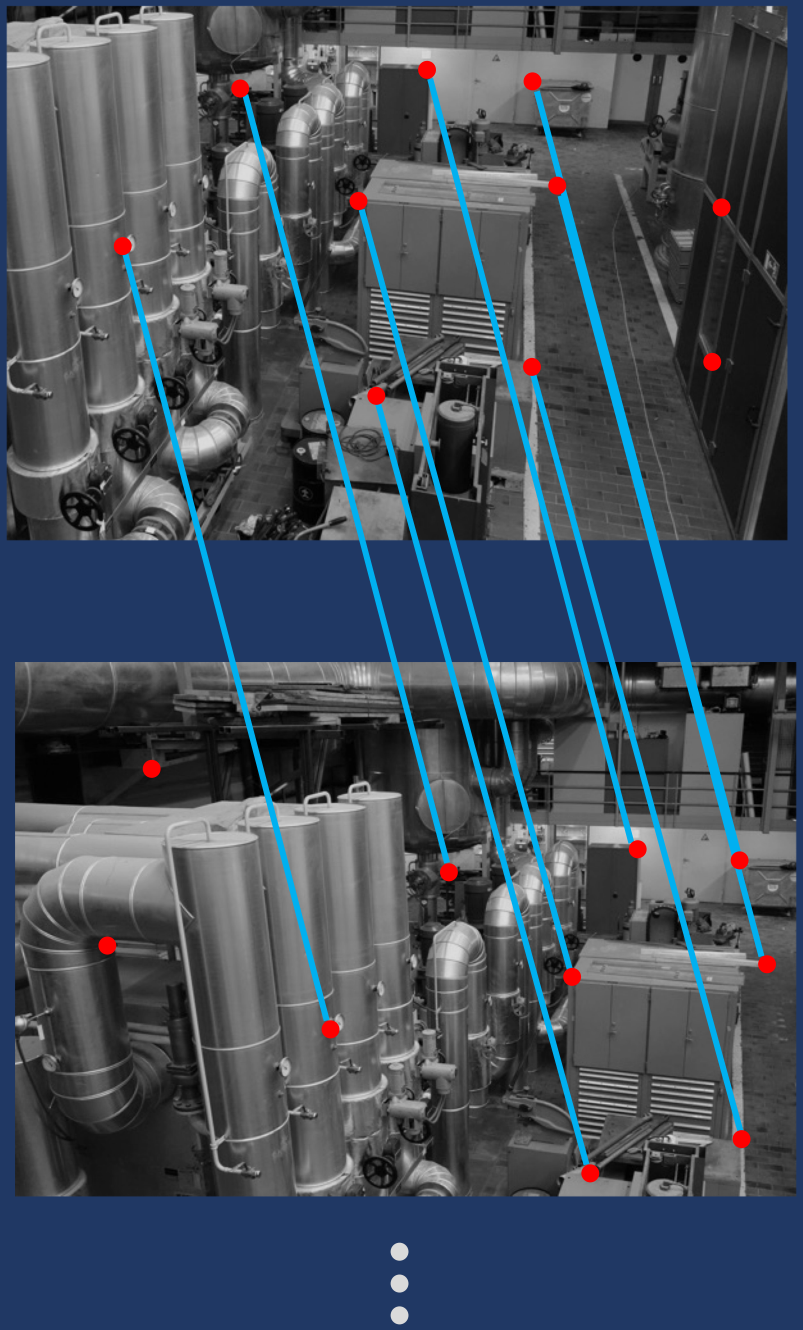
I_{k+1}





I_k

I_{k+1}



Feature-based VO Pipeline

Image Sequence

↓

Feature Detection

↓

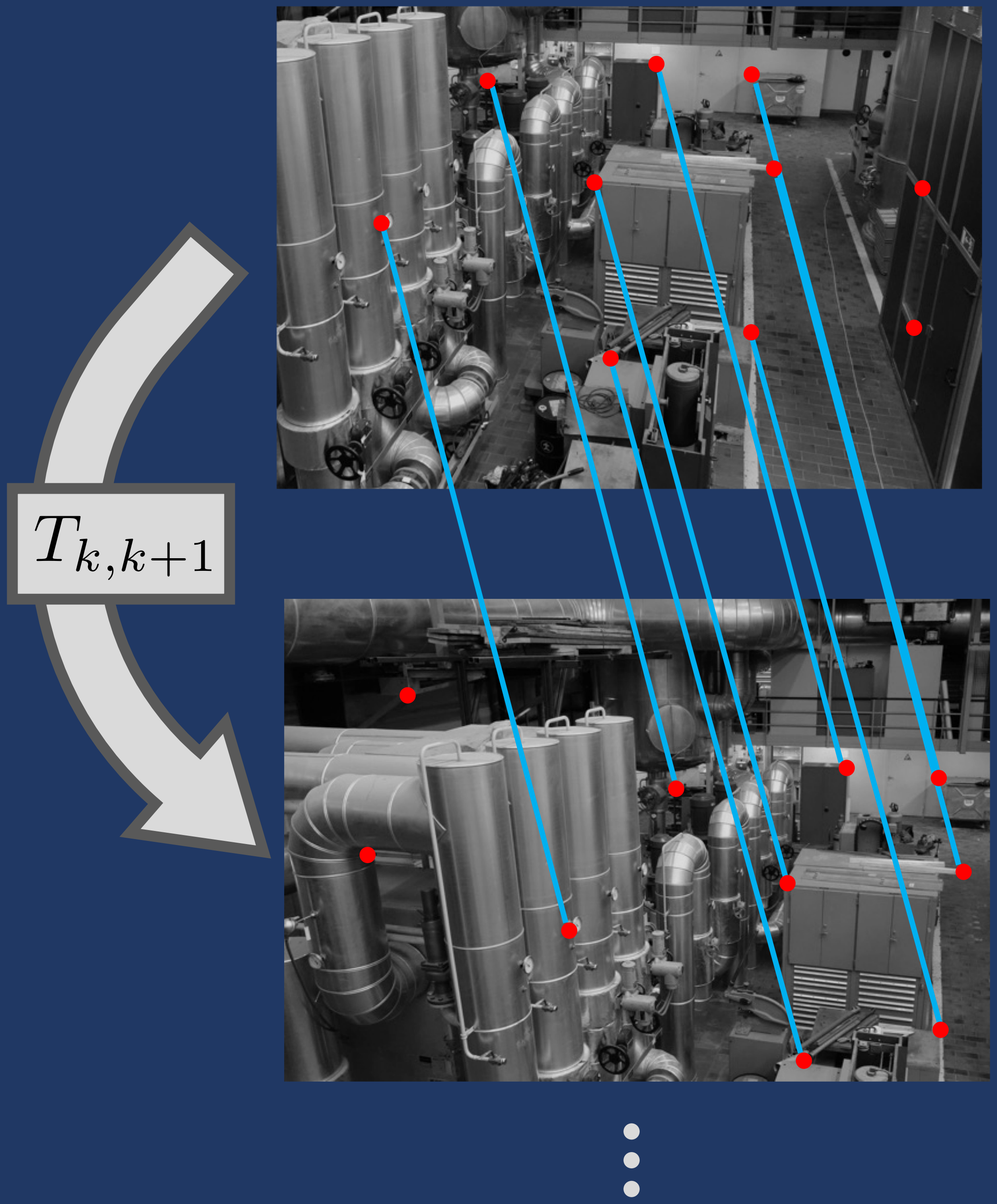
Feature Matching (or Tracking)

↓

Motion Estimation

↓

Local Optimization (Bundle Adjustment)



Feature-based VO Pipeline

Image Sequence

↓

Feature Detection

↓

Feature Matching (or Tracking)

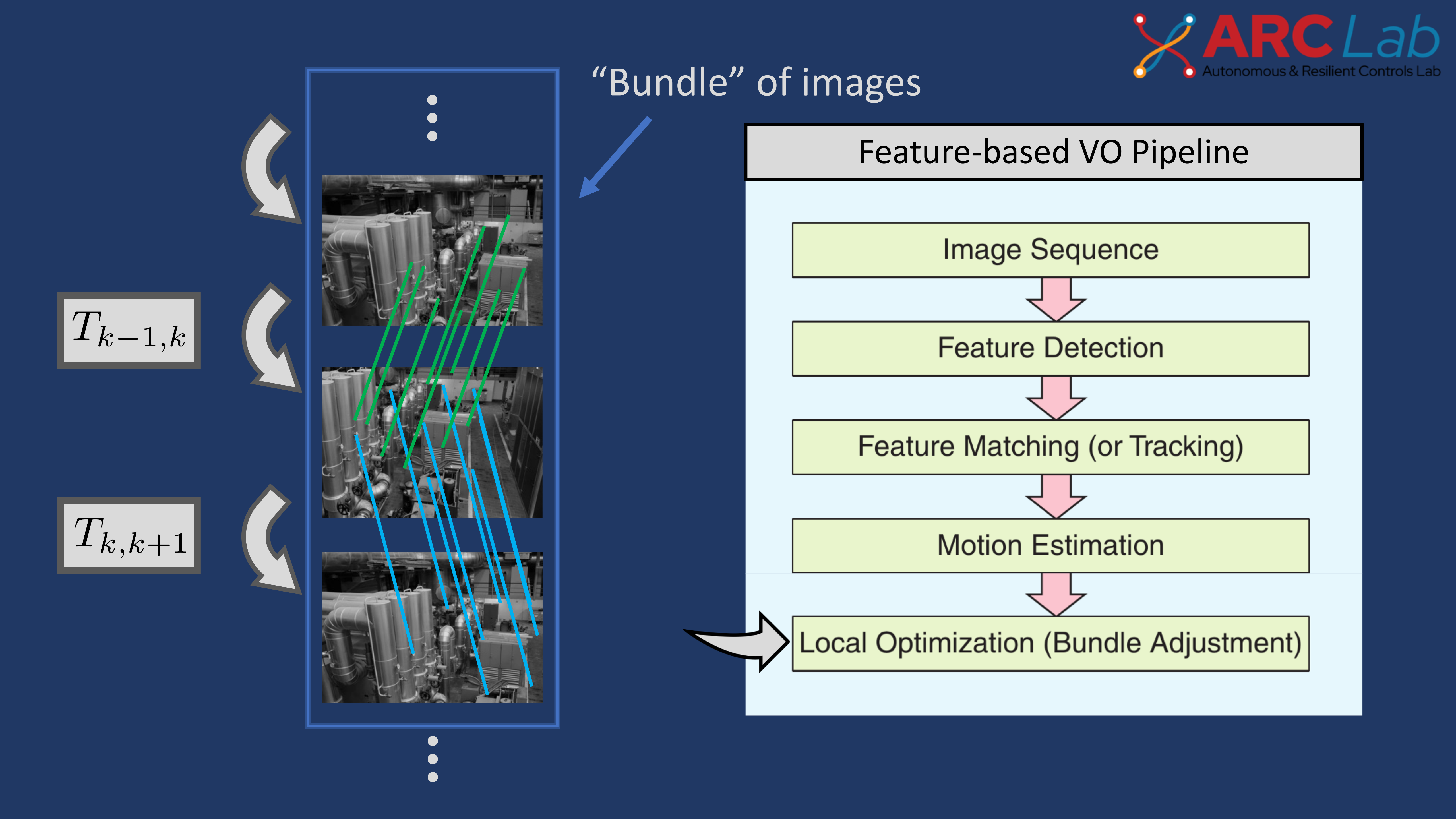
↓

Motion Estimation

↓

Local Optimization (Bundle Adjustment)

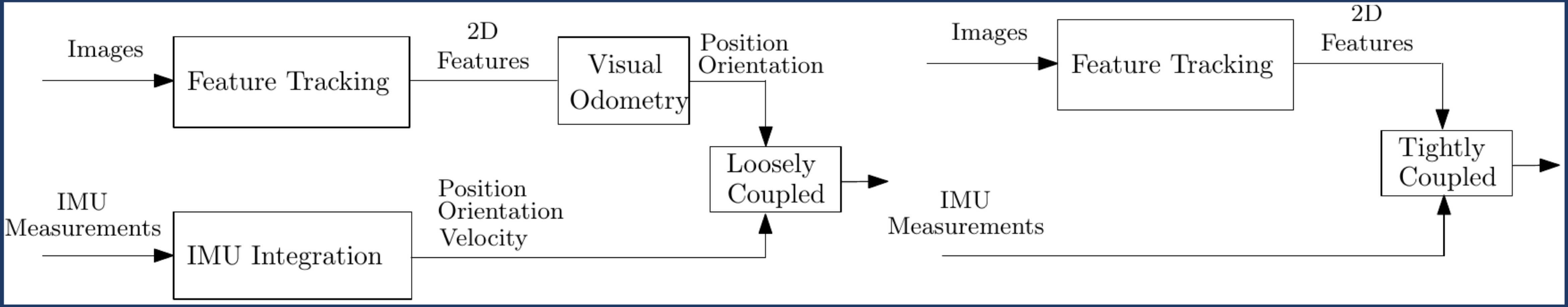
ARC Lab
Autonomous & Resilient Controls Lab


"Bundle" of images
$T_{k-1,k}$
$T_{k,k+1}$
Feature-based VO Pipeline
Image Sequence
Feature Detection
Feature Matching (or Tracking)
Motion Estimation
Local Optimization (Bundle Adjustment)
ARC Lab
Autonomous & Resilient Controls Lab

Visual-inertial odometry (VIO)

The idea: Use one or more cameras and an IMU for pose and velocity estimation.

- VIO better than VO in terms of accuracy and robustness.
- VIO handles fast movements better than VO (good for drones!)
- Tight or loose coupling of IMU measurements and images:



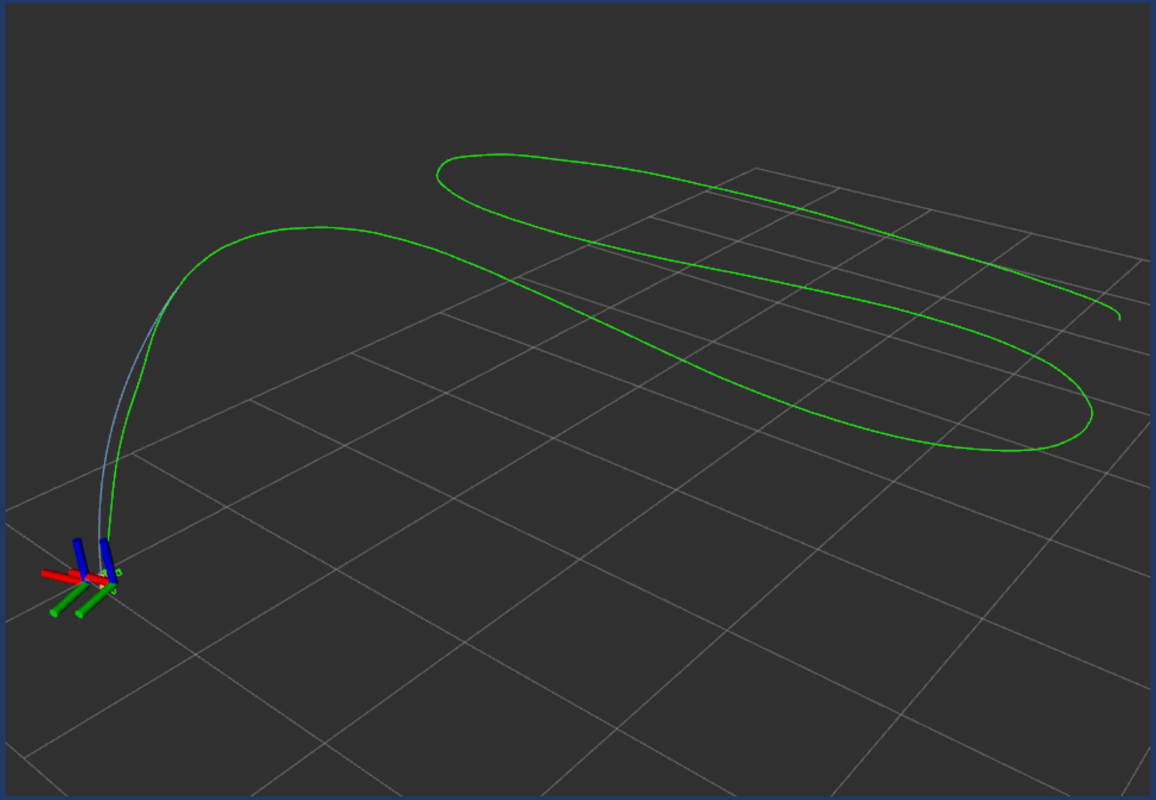
FOV-Constrained Landing

Autonomous & Resilient Controls Lab

Search for the AprilTag marker

The idea: Follow a pre-made search trajectory until the AprilTag is spotted, then represent the AprilTag pose w.r.t. world coordinate frame.

- MPC is used to track the search trajectory.
- Heuristic search trajectory such as zig-zag pattern can be used.



AprilTag marker in world frame

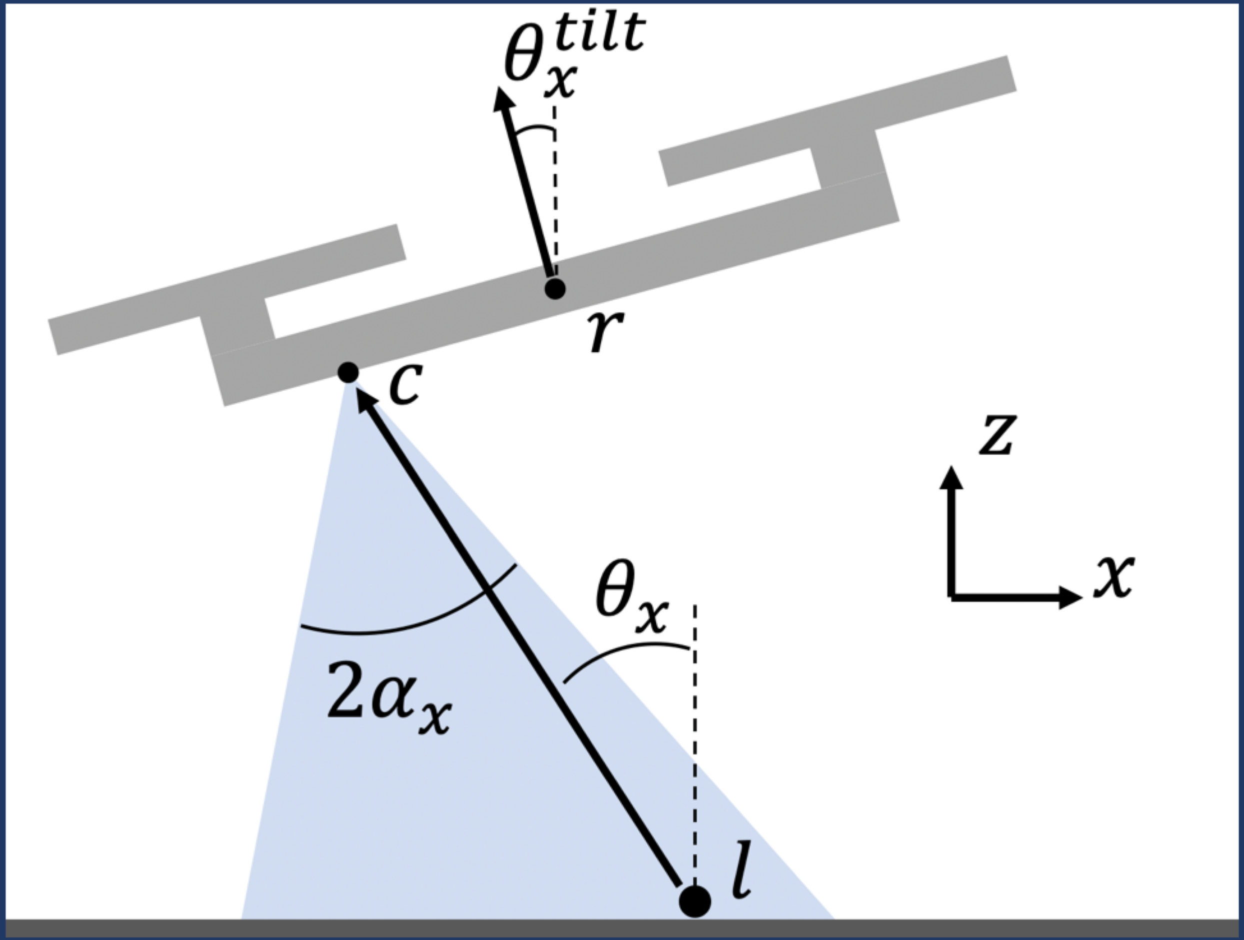
After spotting the AprilTag marker, the AprilTag algorithm estimates its pose w.r.t. camera. We need to estimate AprilTag pose w.r.t world frame.

- T_A^C : AprilTag pose expressed in camera frame.
- T_A^W : AprilTag pose expressed in world frame.
- T_C^B : Camera pose expressed in quadcopter body frame.
- T_B^W : Quadcopter body pose expressed in world frame.

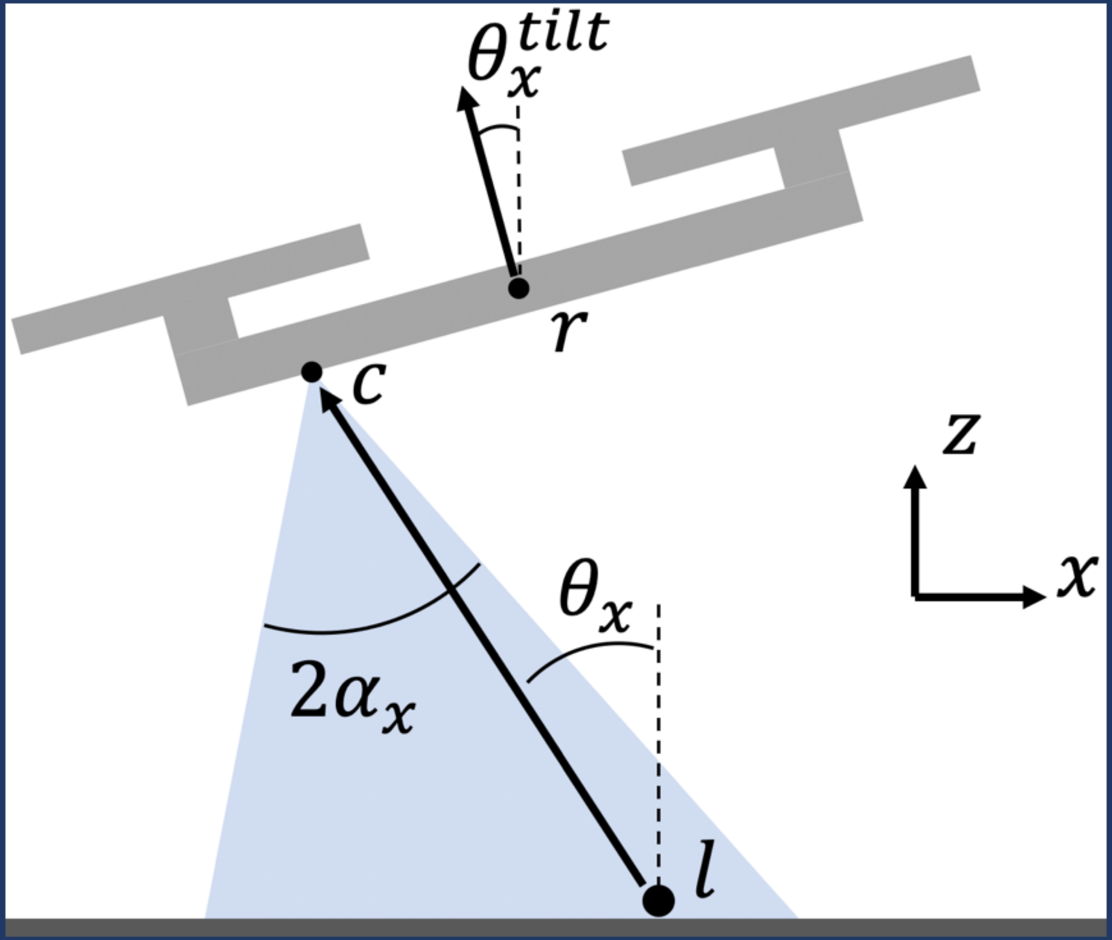
$$\boxed{T_A^W = T_A^C T_C^B T_B^W}$$

FOV constraint

The idea: Construct FOV constraints that can be used in the min-snap QP optimization problem.

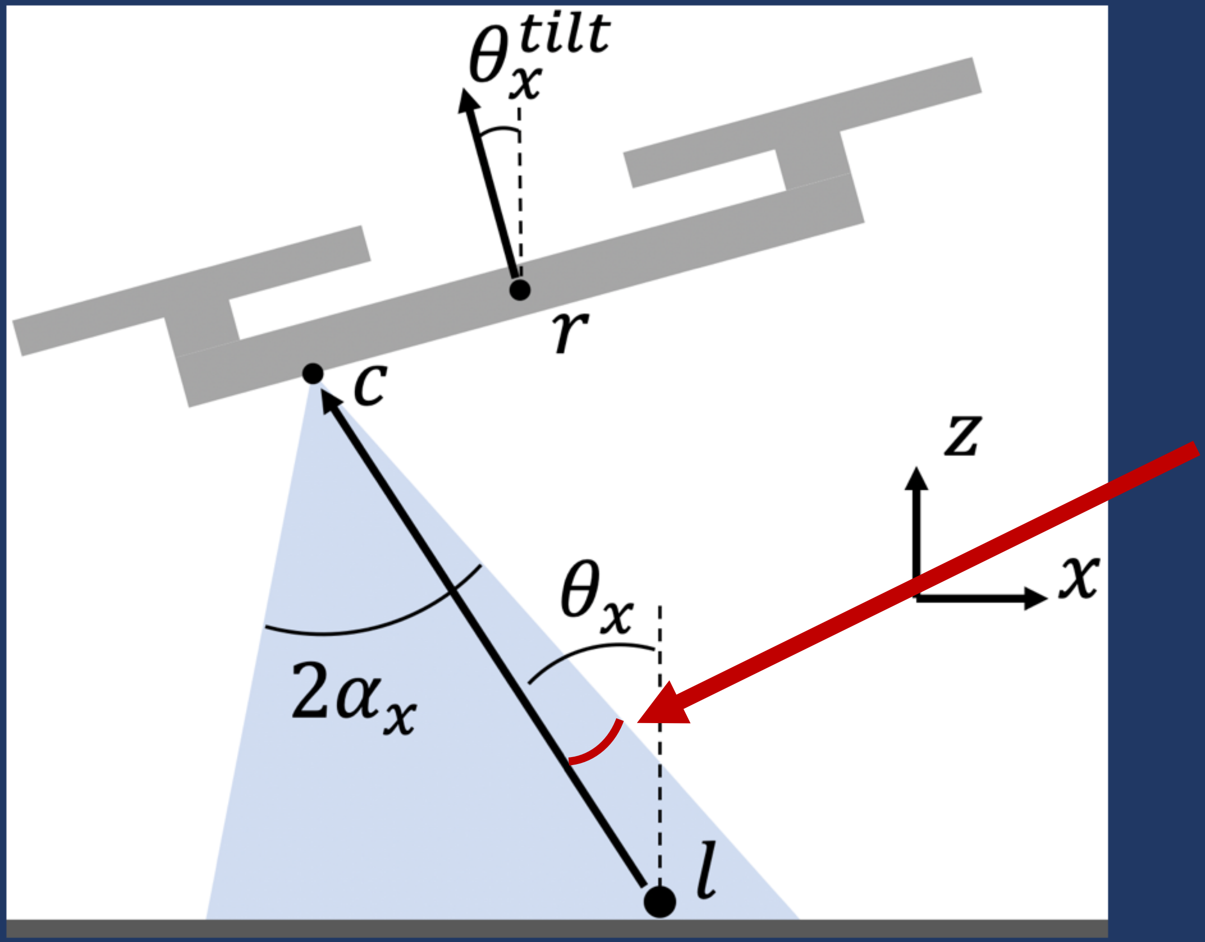


FOV constraint



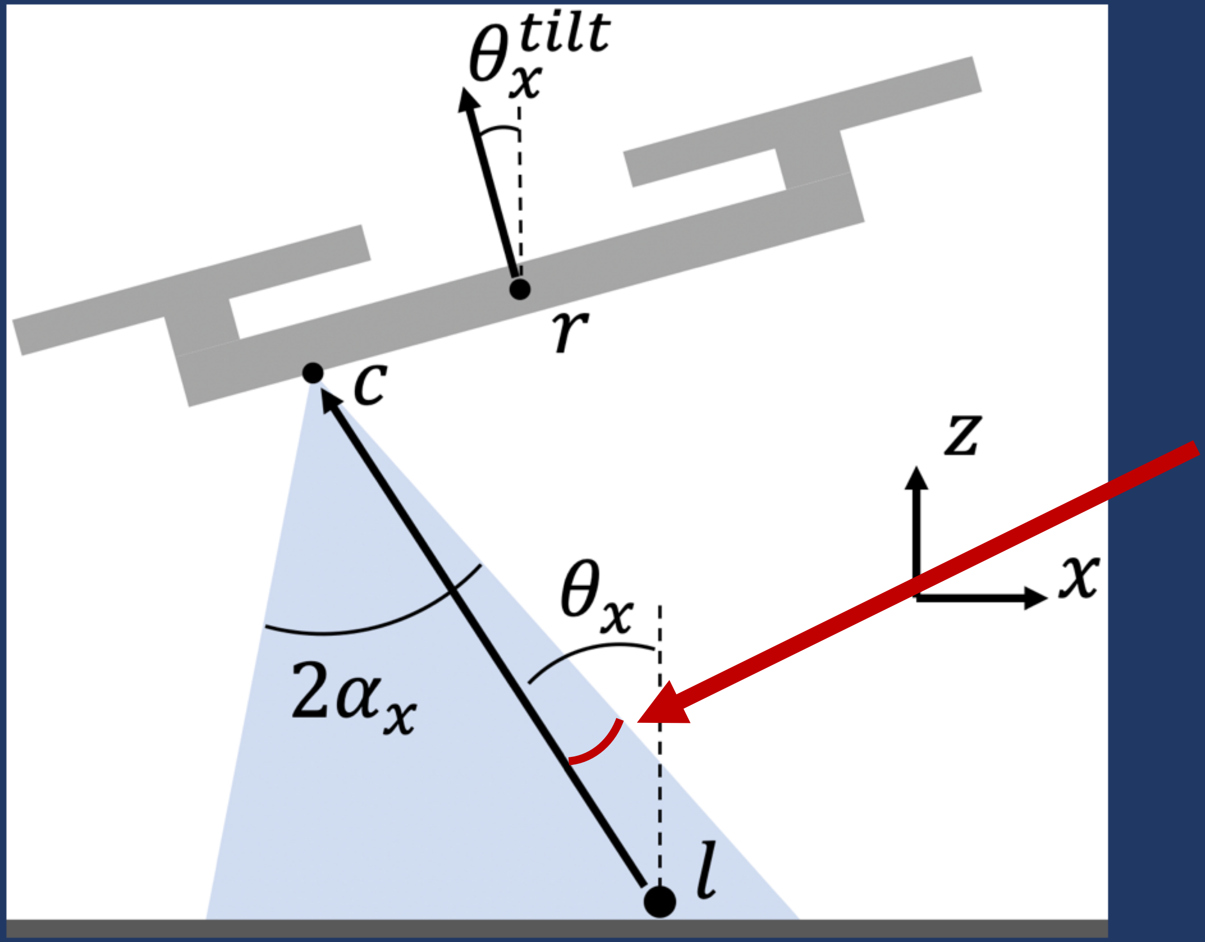
Imagine rotating the quadcopter clockwise about point c.

FOV constraint



Imagine rotating the quadcopter clockwise about point c.

FOV constraint

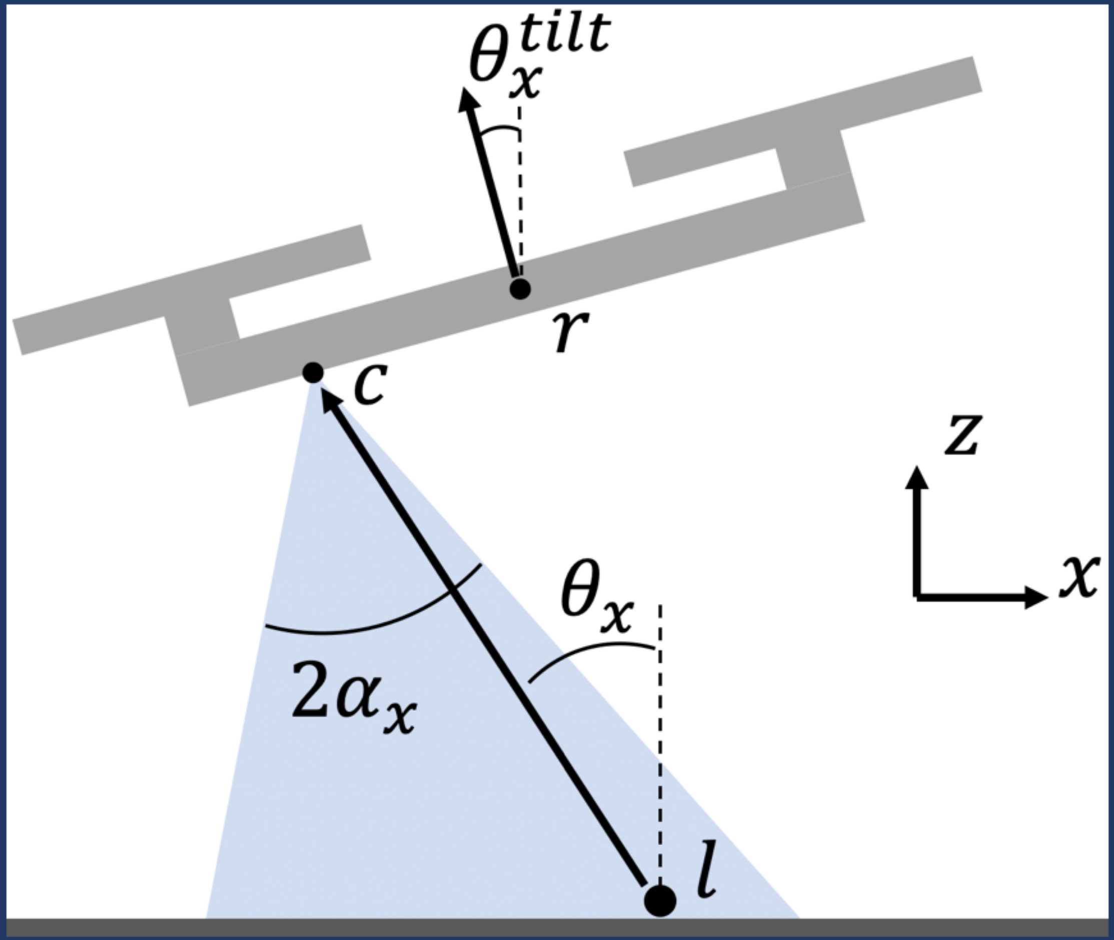


Imagine rotating the quadcopter clockwise about point c.

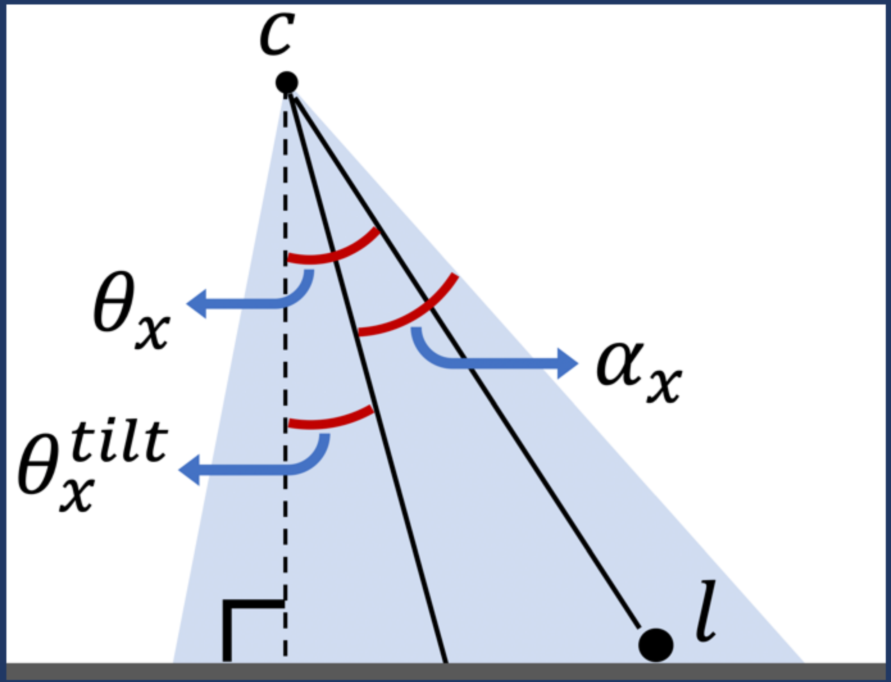
We can only rotate by a *certain amount* before point l leaves the FOV

How do we formalize this constraint?

FOV constraint



After some re-arranging:



In this case, the FOV constraint is:

$$\alpha_x + \theta_x^{tilt} - \theta_x \geq 0$$

FOV constraint

- The general FOV constraint:

$$-(\alpha_x - |\theta_x|) \leq \theta_x^{tilt} \leq \alpha_x - |\theta_x| \qquad (3)$$

⚠ Note (3) needs to be **linear in the min-snap QP decision variables**. Re-writing (3) in terms of acceleration will fix this.

- Planar quadcopter acceleration:

$$a_x = \tau \sin(-\theta_x^{tilt})$$
$$a_z = \tau \cos(\theta_x^{tilt}) - g$$

$$\longrightarrow \quad \frac{-a_x}{a_z + g} = \tan(\theta_x^{tilt}) \qquad (4)$$

FOV constraint

$$-(\alpha_x - |\theta_x|) \leq \theta_x^{tilt} \leq \alpha_x - |\theta_x| \quad (3)$$

$$\frac{-a_x}{a_z + g} = \tan(\theta_x^{tilt}) \quad (4)$$

Note these tangent properties:

$$\tan(0) = 0$$

$$\tan(x) \text{ monotonic increasing, } \forall x \in (\tfrac{-\pi}{2}, \ \tfrac{\pi}{2})$$

Assuming $\theta_x^{tilt} \in (\frac{-\pi}{2}, \frac{\pi}{2})$, taking the tangent of (3) gives:

$$-\tan(\alpha_x - |\theta_x|) \leq \frac{-a_x}{a_z + g} \leq \tan(\alpha_x - |\theta_x|)$$

$$\implies |a_x| \leq \tan(\alpha_x - |\theta_x|)(a_z + g) \quad (5)$$

- By fixing θ_x, (5) is now linear in acceleration, so it can be used in the min-snap QP.

Min-snap QP with FOV constraints

ARC Lab
Autonomous & Resilient Controls Lab

$$\min_{\substack{\boldsymbol{r}_{T_{ij}},\ \psi_{T_{ij}} \\ \forall i \in \{0,\dots,n\} \\ \forall j \in \{1,\dots,m\}}} \int_{t_0}^{t_m} \mu_r \left\| \mathbf{r}_T^{(k_r)}(t) \right\|^2 + \mu_\psi \left(\psi_T^{(k_\psi)}(t) \right)^2 dt \quad (6.1)$$

boundary conditions

$$\text{s.t.} \quad \mathbf{r}_T^{(p)}(t_j) = \mathbf{r}_j^{(p)} \text{ or free}, \quad j = \{0, m\};\ p = 0, \dots, k_r \quad (6.2)$$

$$\psi_T^{(p)}(t_j) = \psi_j^{(p)} \text{ or free}, \quad j = \{0, m\};\ p = 0, \dots, k_\psi \quad (6.3)$$

continuity

$$\sum_{i=p}^{n} (\boldsymbol{r}_{T_{ij}} - \boldsymbol{r}_{T_{i,j+1}}) = 0, \quad j = 1, \dots, m-1;\ p = 0, \dots, k_r \quad (6.4)$$

$$\sum_{i=p}^{n} (\psi_{T_{ij}} - \psi_{T_{i,j+1}}) = 0, \quad j = 1, \dots, m-1;\ p = 0, \dots, k_\psi \quad (6.5)$$

FOV constraints

$$|a_x(t_j)| \le \tan(\alpha_x - |\theta_x(t_j)|)(a_z(t_j) + g), \quad j = 1, \dots, m-1 \quad (6.6)$$

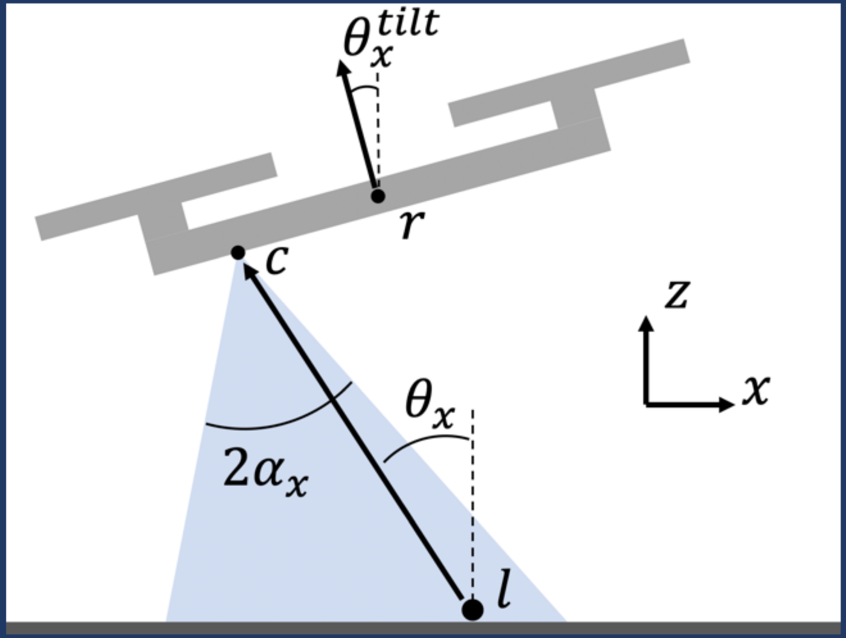
$$|a_y(t_j)| \le \tan(\alpha_y - |\theta_y(t_j)|)(a_z(t_j) + g), \quad j = 1, \dots, m-1 \quad (6.7)$$

$$[a_x(t), a_y(t), a_z(t)]^T := \boldsymbol{r}_T^{(2)}(t)$$

Min-snap QP with FOV constraints

How to chose $\theta_x(t_j),\ \forall j = 1, \ldots, m-1$ from (6.6)?

- Both $\theta_x(t_0)$ and $\theta_x(t_m)$ are known:
- $\theta_x(t_0) = \tan^{-1}\left(\frac{l_x(t_0) - c_x(t_0)}{c_z(t_0) - l_z(t_0)}\right)$
- $\theta_x(t_m) = 0$
- We then heuristically choose evenly-spaced $\theta_x(t_j),\ \forall j = 1, \ldots, m-1$ values between $\theta_x(t_0)$ and $\theta_x(t_m)$.
- The argument follows for $\theta_y(t_j)$.



Check feasibility of FOV constraints

$$|a_x(t_j)| \leq \tan(\alpha_x - |\theta_x(t_j)|)(a_z(t_j) + g), \quad j = 1, \ldots, m-1 \qquad (6.6)$$
$$|a_y(t_j)| \leq \tan(\alpha_y - |\theta_y(t_j)|)(a_z(t_j) + g), \quad j = 1, \ldots, m-1 \qquad (6.7)$$

Since $|\theta_x(t_0)| \geq |\theta_x(t_i)|, \; \forall i = 1, \ldots, m$ as previously determined,

and we assume $a_z \geq -g$ (can't accel. down faster than gravity),

then if $\alpha_x - |\theta_x(t_0)| > 0$ is satisfied, (6.6) is feasible. Same goes for (6.7)

In practice, we check the feasibility of (6.6) and (6.7) before attempting to solve the min-snap QP

PAMPC tracking control

The idea: MPC control with an extra perception cost term in the objective function.

- The perception cost function is: $\mathbf{s}(t) - \mathbf{s}_d$, where $\mathbf{s}(t)$ is the projection of AprilTag into image plane, and $\mathbf{s}_d$ is the principal point

$$\mathbf{s}(t) := [u_A(t) \;\; v_A(t)]^T \qquad \mathbf{s}_d := [c_x \;\; c_y]^T$$

$$u_A(t) = f_x \frac{[\boldsymbol{r}_A^C(t)]_x}{[\boldsymbol{r}_A^C(t)]_z} + c_x, \quad v_A(t) = f_y \frac{[\boldsymbol{r}_A^C(t)]_y}{[\boldsymbol{r}_A^C(t)]_z} + c_y$$

- $\boldsymbol{r}_A^C(t)$ is position from camera to AprilTag. Computation found in [2]

[2]: Falanga et al. "PAMPC: Perception-Aware Model Predictive Control for Quadrotors". 2018

PAMPC tracking control

$$\boldsymbol{z}_N := \begin{bmatrix} \boldsymbol{x}(N) - \boldsymbol{x}_d(N) \\ \mathbf{s}(N) - \mathbf{s}_d \end{bmatrix}, \quad \boldsymbol{z}(t) := \begin{bmatrix} \boldsymbol{x}(t) - \boldsymbol{x}_d(t) \\ \mathbf{s}(t) - \mathbf{s}_d \\ \boldsymbol{u}(t) \end{bmatrix}.$$

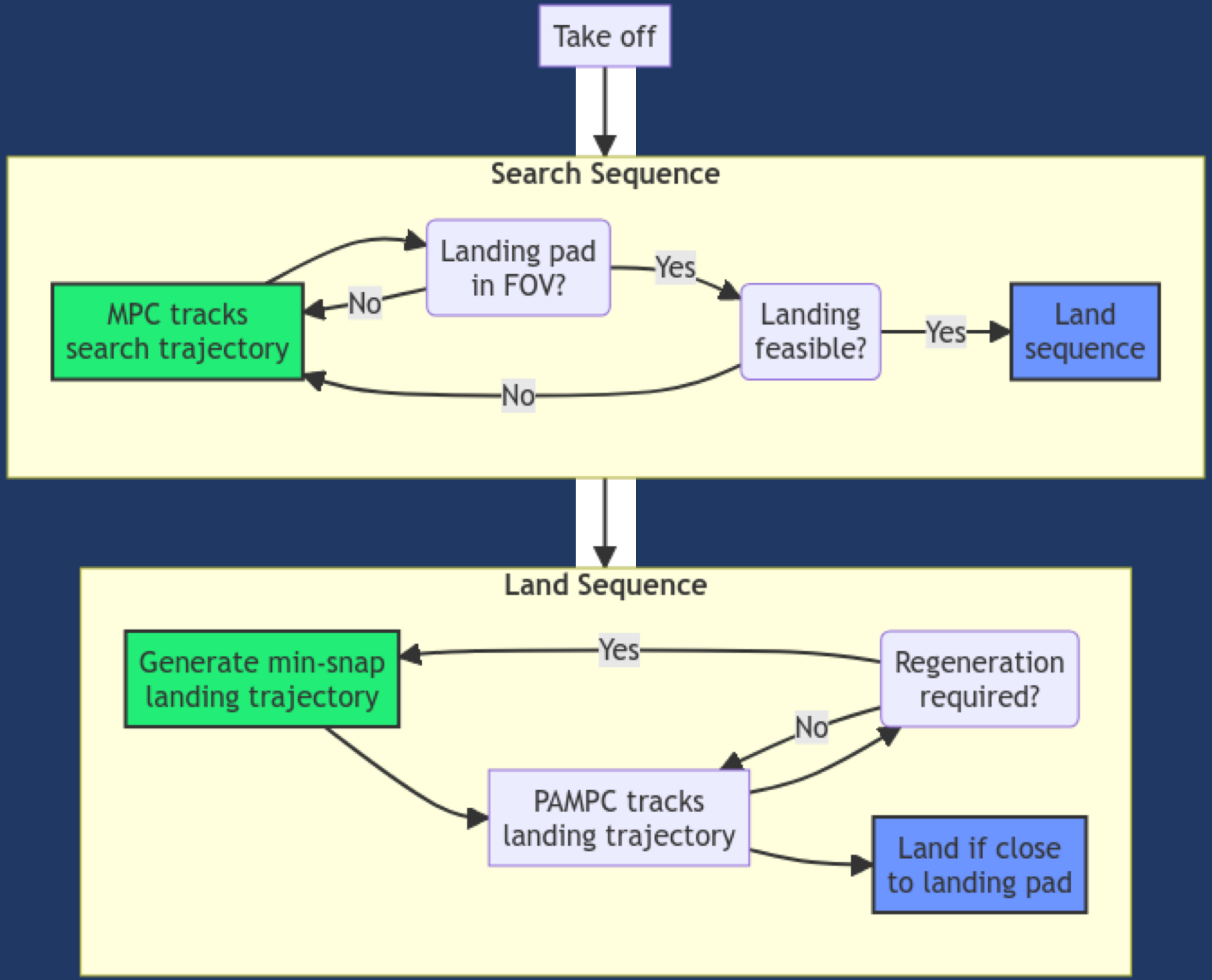
$$\begin{aligned} \min_{\substack{\boldsymbol{x}(t_1), \dots, \boldsymbol{x}(t_N) \\ \mathbf{s}(t_1), \dots, \mathbf{s}(t_N) \\ \boldsymbol{u}(t_0), \dots, \boldsymbol{u}(t_{N-1})}} \quad & \boldsymbol{z}_N^T Q_N \boldsymbol{z}_N + \sum_{i=0}^{N-1} \boldsymbol{z}(t_i)^T Q \boldsymbol{z}(t_i) \\ \text{s.t.} \quad & \boldsymbol{x}(t_0) = \hat{\boldsymbol{x}}(t_0) \\ & \mathbf{s}(t_0) = \hat{\mathbf{s}}(t_0) \\ & \boldsymbol{x}(t_i) = F(\boldsymbol{x}(t_{i-1}), \boldsymbol{u}(t_{i-1})), \quad \forall i \in \{1, \dots, N\} \\ & \mathbf{s}(t_i) = [u_A(\boldsymbol{x}(t_i)),\ v_A(\boldsymbol{x}(t_i))]^T, \quad \forall i \in \{1, \dots, N\} \\ & \boldsymbol{u}(t_{i-1}) \in \mathcal{U}, \quad \forall i \in \{1, \dots, N\} \end{aligned}$$

PAMPC tracking control

Note:

While our min-snap QP provides an FOV-constrained trajectory, implementing the PAMPC adds a layer of redundancy for improved visibility of the landing pad.

Flight and landing logic

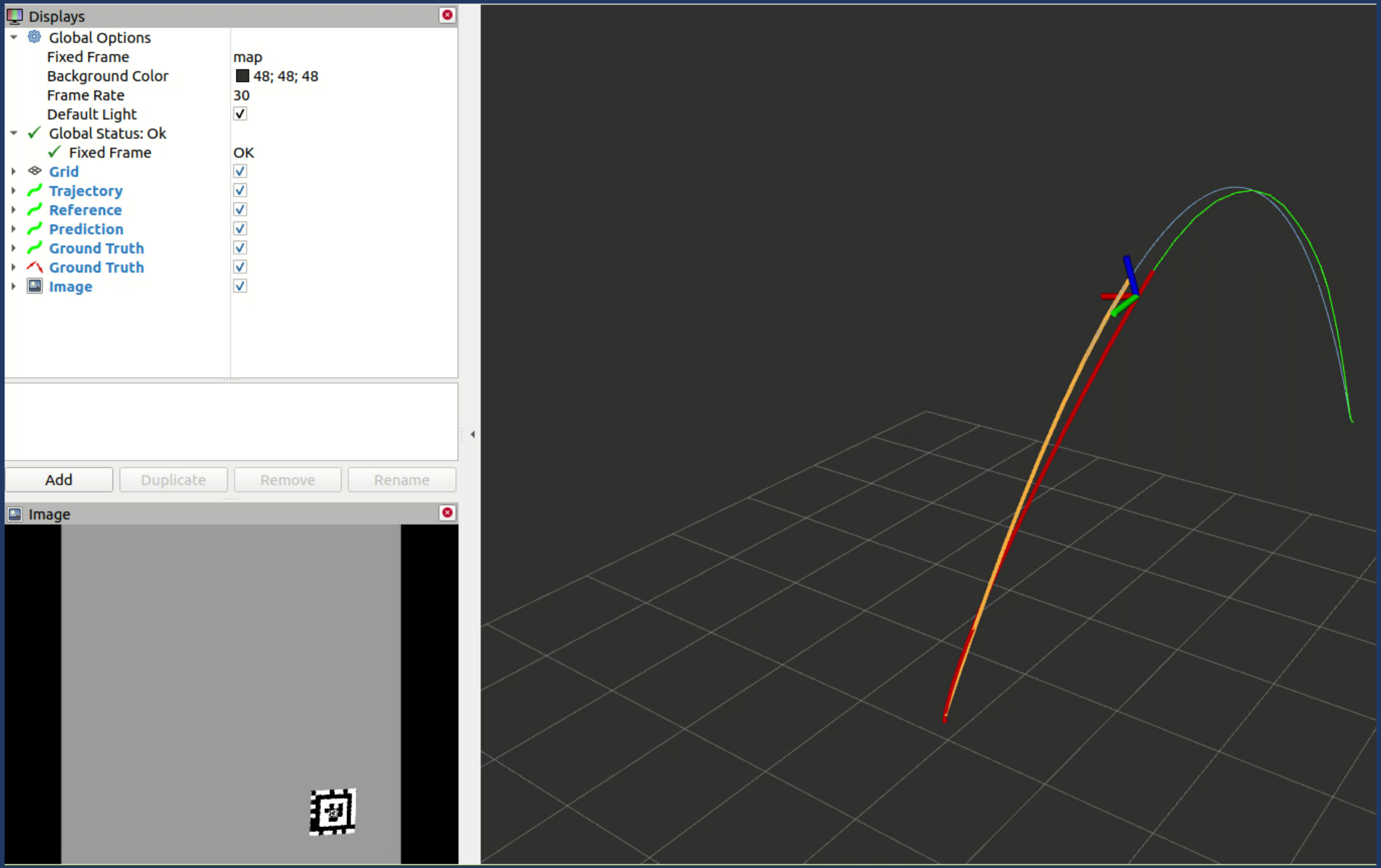


Evaluation

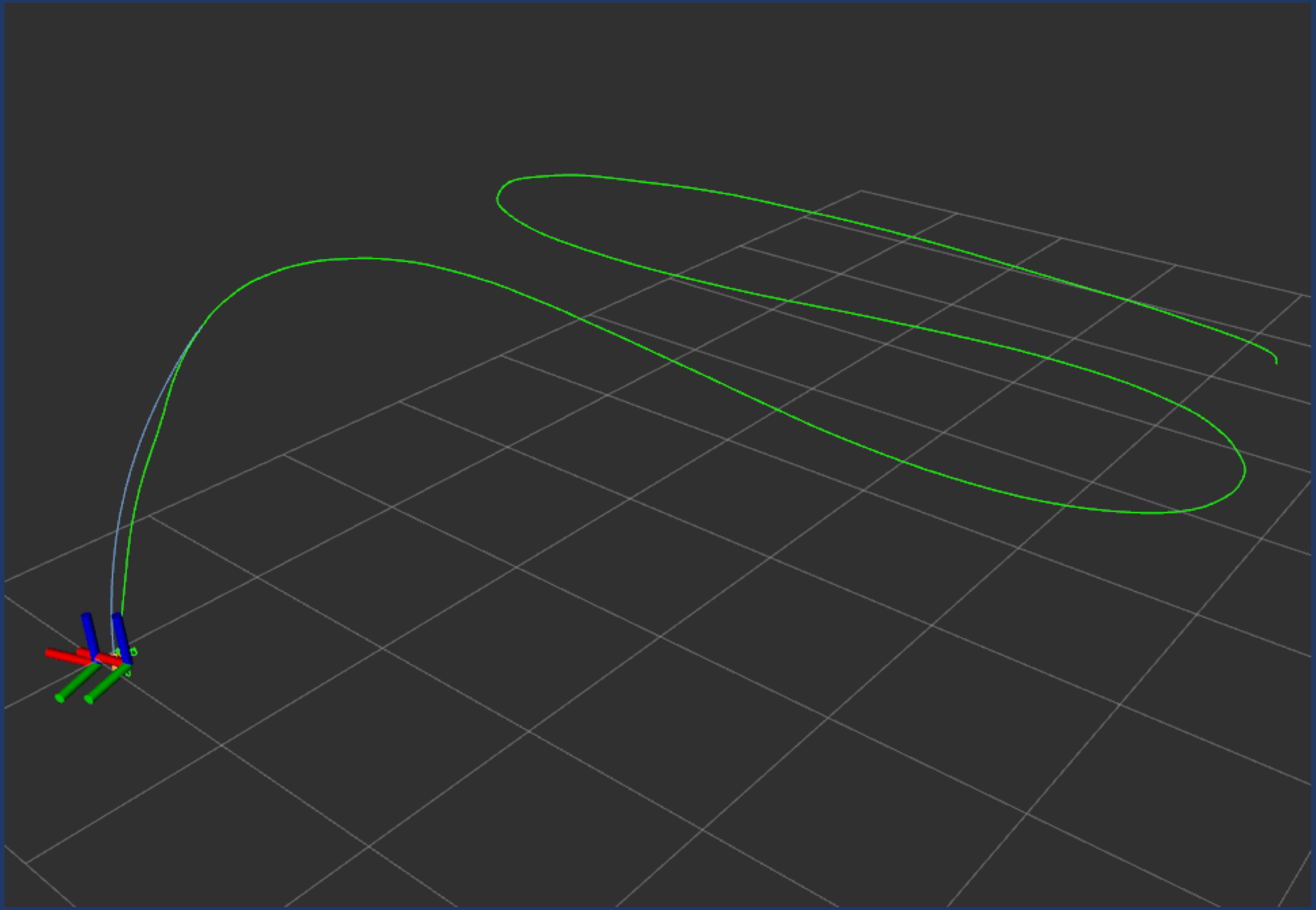
Experiments

ARC Lab
Autonomous & Resilient Controls Lab

Experiments



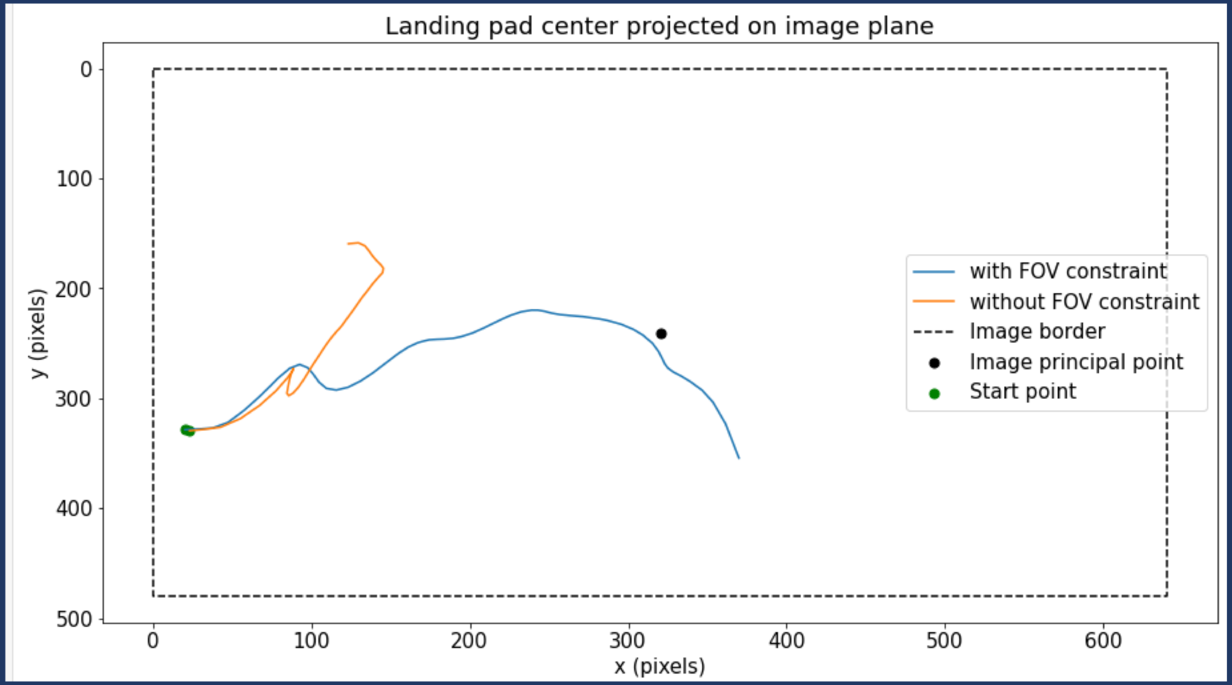
Experiments

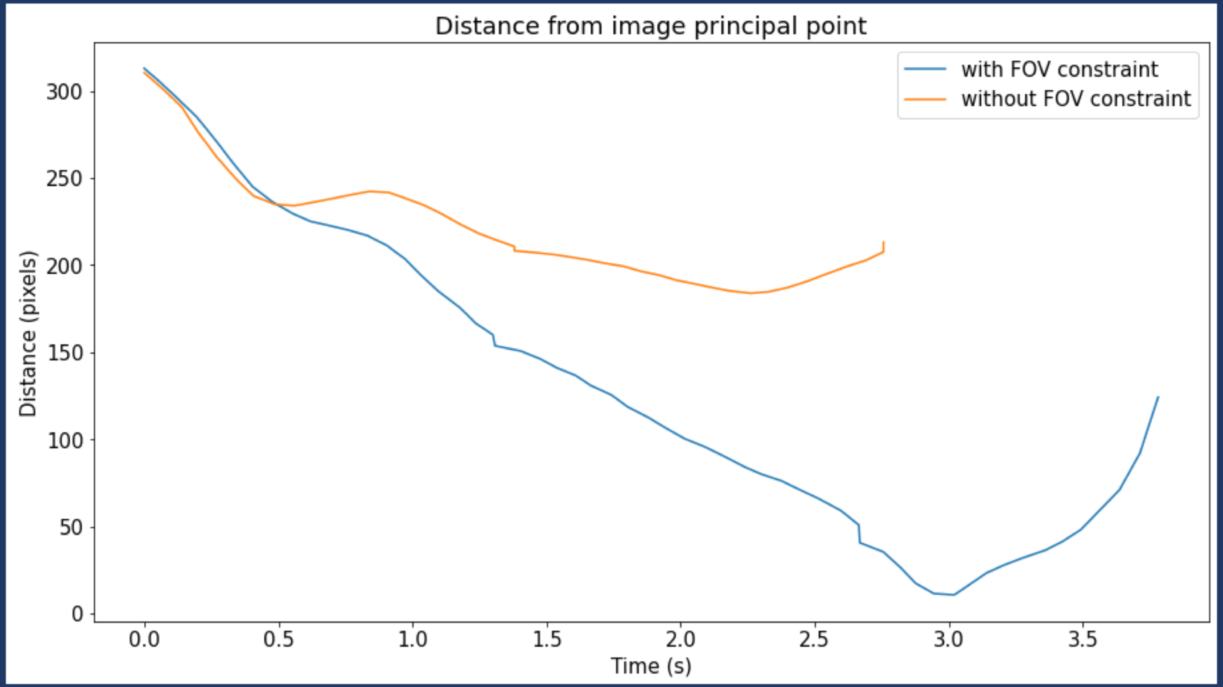


Autonomous & Resilient Controls Lab

Experiments

Results





We can see that the FOV constraint allows the projected landing pad center to get closer to the principal point when compared to the same trajectory without FOV constraints.

Questions?