MANCHESTER
1824
The University of Manchester

Thesis Title

A thesis submitted to the University of Manchester for the degree of
Doctor of Philosophy
in the Faculty of Science and Engineering

2022

Damian Crosby
Department of Mechanical, Aerospace and Civil Engineering

Contents

Word Count: X

List of figures

List of tables

List of publications

Publications go here.

List of abbreviations

AUJ Actuated Universal Joint

2D Two Dimensional

3D Three Dimensional

COM Centre of Mass

DLS Damped Least Squares

DOF Degree(s) of Freedom

PID Proportional Integral Derivative

SHTP Sensor Hub Transport Protocol

MSB Most Significant Bit

LSB Least Significant Bit

I^2C Inter-Integrated Circuit

SPI Serial Peripheral Interface

UART Universal Asynchronous Receiver-Transmitter

DIO Digital Input/Output

IMU Inertial Measurement Unit

GOOP Graphical Object Oriented Programming

CS Chip Select

Abstract

This is abstract text.

Declaration of originality

I hereby confirm that no portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright statement

i The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made *only* in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=24420`), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see `http://www.library.manchester.ac.uk/about/regulations/`) and in The University's policy on Presentation of Theses.

Acknowledgements

Acknowledgements go here.

Chapter 1

A Literature Review of Terrestrial and Static Robots with Robotic Tails and Their Functions

1.1 Introduction

1.2 Research Methodology

1.3 Tail Functions of Terrestrial Robots

Here "surface" is strictly defined as a two-dimensional movement manifold which the robot can reach while being in constant contact with the manifold.

Many terrestrial robots have a tail that is only partially active, in particular

Three different tiers of classification.

1. **Environment:** The general domain the robotic system is operating in when the tail is active. From the reviewed literature, three categories have been created:
    - *Terrestrial:* A robot with an active tail when the robot is touching a surface, such as a robot driving along the ground.
    - *Aerial:* A robot with an active tail when the robot is in free space, such as a robot which *has jumped* into the air, or is falling off a ledge.
    - *Transition:* A robot with an active tail when the robot is just about to transition from between the two previous environments, such as a robot *just about* to jump into the air.
2. **Action:** The specific action the robot is performing when the tail is active. Most actions are unique to each environment, except for *hopping*.
    - *Straight:* A robot travelling across a surface maintaining its direction of travel.

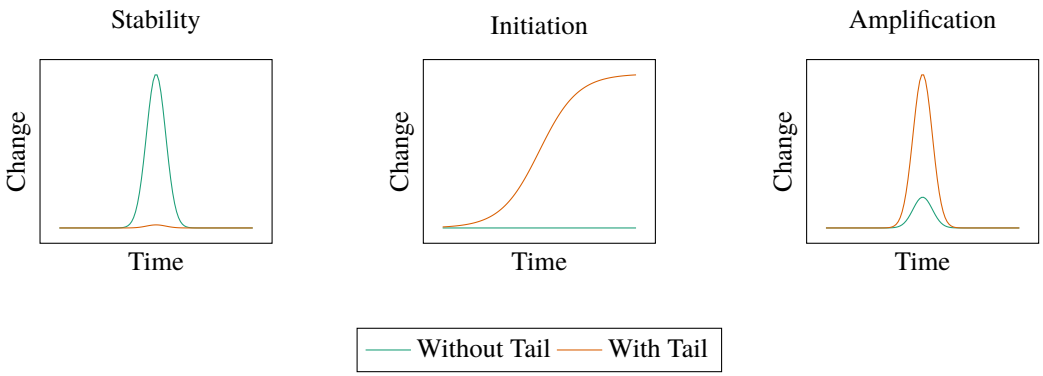


Figure 1.1: Abstract graphs of the different functions of the tail, with the magnitude representing some kind of change in the robot's position and/or orientation.

- *Accelerating:* A robot changing its velocity in the direction of travel across a surface. In the literature, this was a robot coming to complete stop, and starting from stationary.
- *Turning:* A robot changing its direction of travel across a surface, such as a robot turning a corner.
- *Balancing:* A robot undergoing external disturbances while travelling across a surface, typically due to adverse terrain, such as a robot navigating a rough and uneven surface without falling over.
- *Hopping:* A robot executing a sequence of periodic jumps in order to travel across a surface, similar to the method of locomotion of a Kangaroo.
- *Jumping:* A robot executing isolated non-periodic jumps, typically to transition from one surface to another at a different altitude and/or orientation.
- *Falling:* A robot transitioning from one surface to another at a different altitude via the influence of gravity.

3. **Function:** The purpose of the tail when the robot is performing the action. These categories usually apply to multiple actions.

   - *Stability:* The tail is used to *maintain* some aspect of the robot's position and/or orientation from the start of the action.
   - *Initiation:* The tail is used to *change* some aspect of the robot's position and/or orientation from the start of the action.
   - *Amplification:* The tail is used to *amplify* the effects of an action by other parts of the robot (such as the legs) which changes the position and/or orientation.

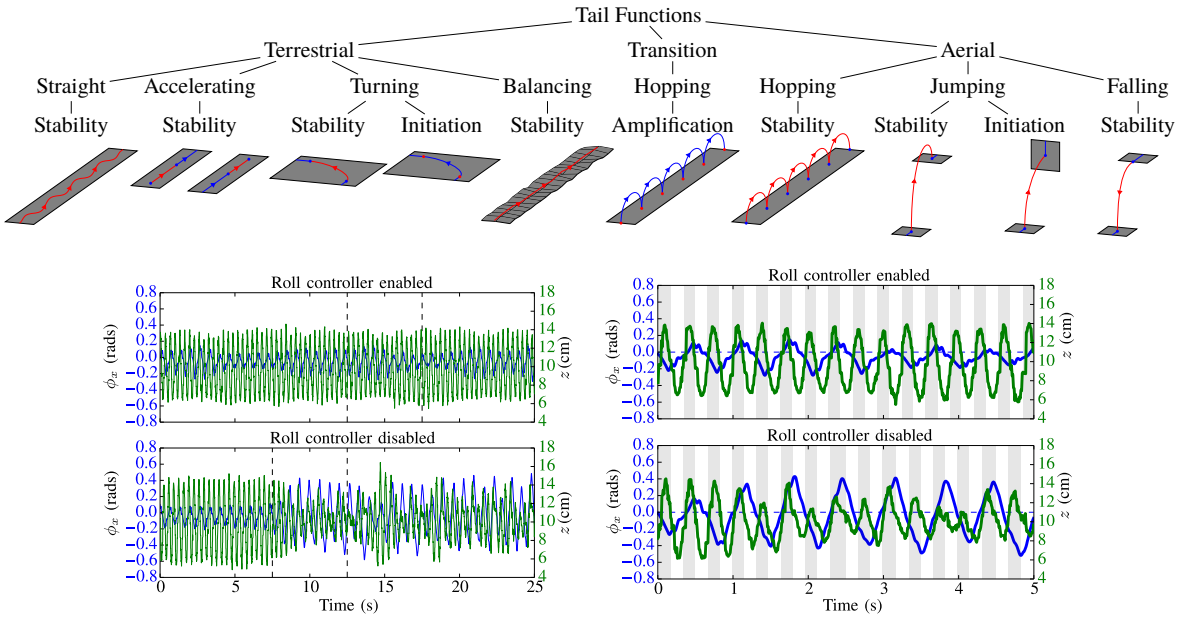


Figure 1.2: Graphs from [1] showing a steady state hopping gait with and without the tail roll controller. Body roll angle ϕ_x is in blue, and body height z is in green. These graphs demonstrate without a roll controller ϕ_x quickly increases in magnitude and becomes chaotic.

1.3.1 Angular Stability during Locomotion

1.3.2 Angular Stability in Free Space

use a tail to change the orientation of a robot (by imparting a torque on the body) when the robot is in "free space" i.e. no part of the robot is in contact with the ground or other external surface. This is the most common use of a tail in terrestrial robots.

Hopping/Galloping Gait

[1] use a tail when in the aerial phase of a hopping/galloping gait. This is to compensate for uneven terrain and/or slight differences in forces generated by each leg when in the leaping phase, imparting a pitch or roll moment on the body.

Single Jump

[2], [3] use a tail

In [2], a wheeled robot with a 1 DOF tail is compared to a lizard (*Agama agama*) that uses its tail to minimise pitch rotation after leaping off a surface. The robot outperformed the lizard, as can be seen in figure **??**.

In [3], a legged robot with a 1 DOF tail is able to change the body orientation so it lands on its side in a more stable configuration, then using the tail to self right.

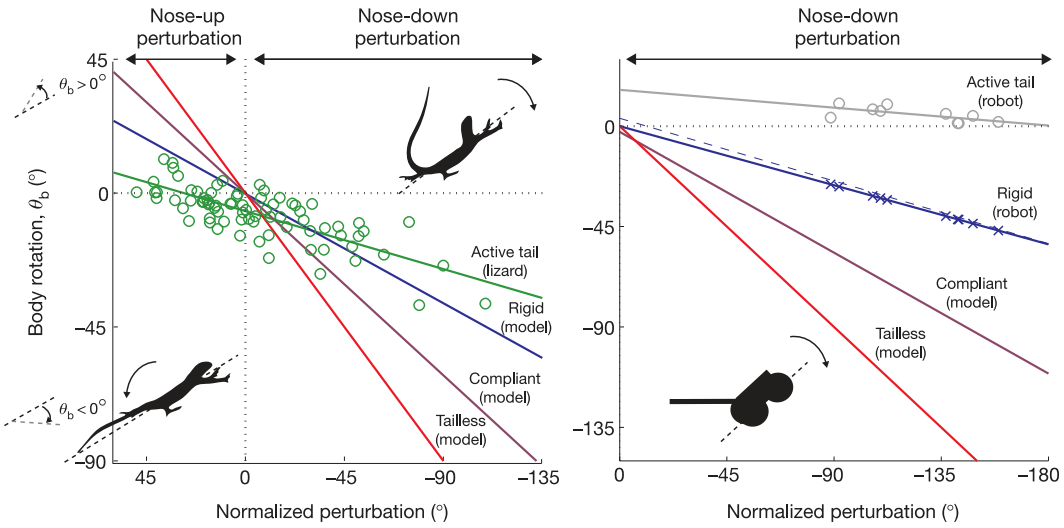


Figure 1.3: Graphs from [2] showing how a lizard and a robot can counteract a "normalised pertubation" disturbance, defined by how much the body would rotate without a tail.

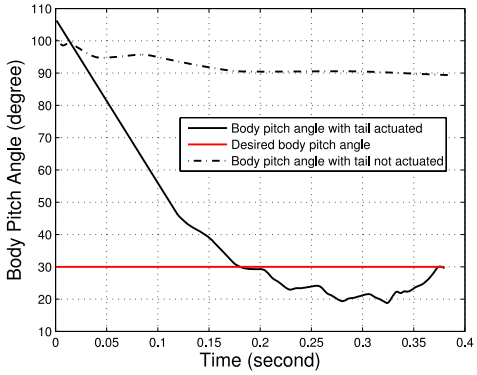


Figure 1.4: Graph from [3] showing the desired body pitch angle for landing on its side, and the trajectory of a jump with an actuated and unactuated tail.

Free Fall

1.3.3 Angular Stability after Disturbance

[4], [5] use a tail to prevent a quadrupedal robot tipping when a disturbance force is imparted (using a "wrecking ball" hitting the body of the robot).

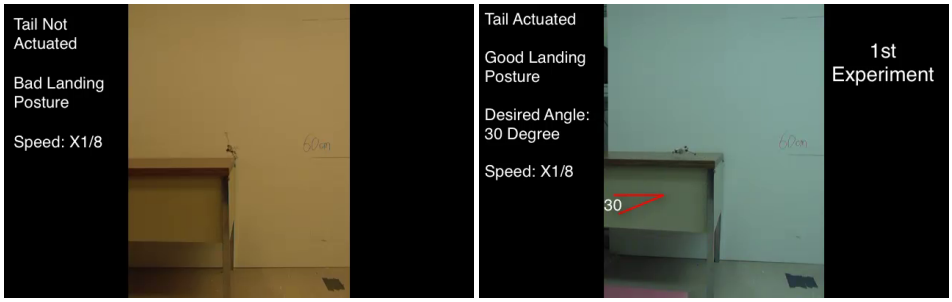


Figure 1.5: Still images from [3] showing an unstable landing on its feet, and a stable landing on its side.

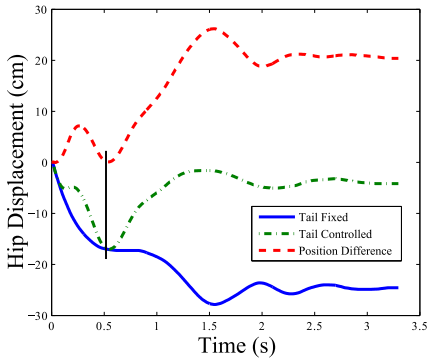


Figure 1.6: Graph from [4] showing how an active tail can minimize body sway when a disturbance force is imparted.

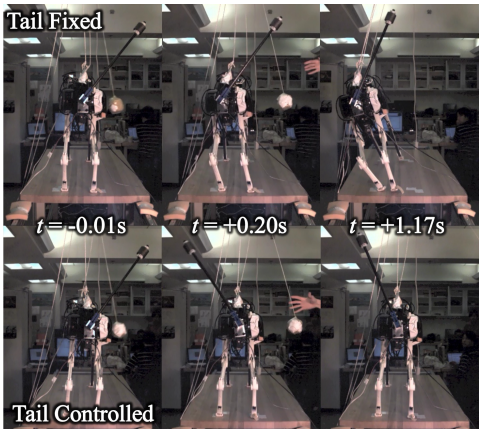


Figure 1.7: Still images from [4] showing the robot absorbing the impact of the wrecking ball with and without an active tail.

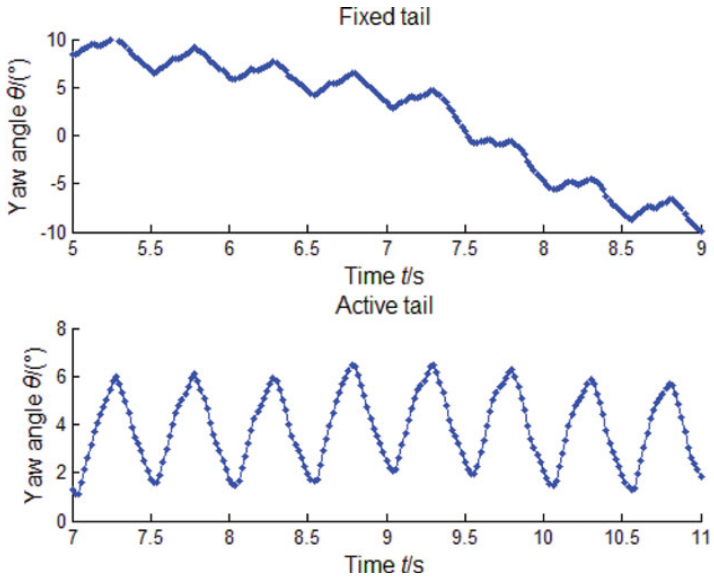


Figure 1.8: Data from [6] showing the effects of a static and dynamic tail on maintaining robot yaw angle in a walking gait.

1.3.4 Path Drift

A number of papers with different kinds of locomotion

Inertial

[6] uses a quadrupedal robot with a 1 DOF head and tail. By swinging the tail on the roll axis in synchronization with the leg motion, but in such a way to counteract the unwanted leg dynamics, the robots yaw angle (heading) was stabilised, as can be seen in figure **??**.

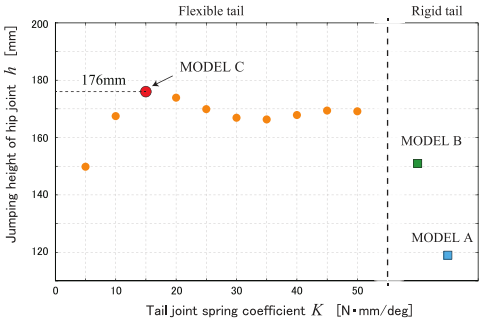


Figure 1.9: Graph from [8] showing how by optimising the spring constant for the spring joints a maximum jump height can be achieved.

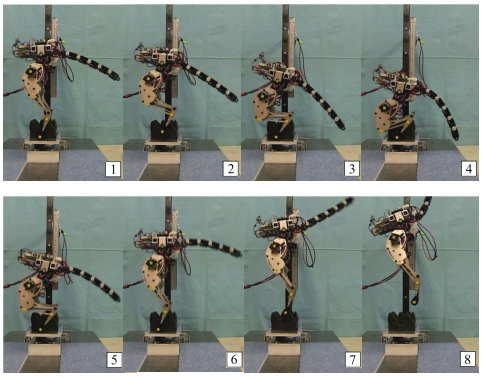


Figure 1.10: Still images from [8] showing the whip-like trajectory of the tail in flight.

Ground Contact

1.3.5 Jumping Height

use a tail to increase the apex of a jump. This can be done either by imparting a dynamic force on the robot body (*Inertial*) or striking the ground (*Ground Contact*). This is generally done in sync

Inertial

[7], [8] developed an open-loop system that synchronised with a hopping cycle using only a single actuator. [7] used a rotating disk to adjust tension on 4 wires terminating at each segment of a 4 segment nylon tail with an accordion pattern to allow deformation, while [8] used a single actuator at the base of the tail, and adjustable bidirectional spring joints on 5 additional segments to create a whip-like trajectory.

While [7] failed to increase the jumping height of the robot (attributed to a low tail mass), [8] was able to successfully prove a dynamic tail could increase the jumping height of a robot.

Ground Contact

[9] uses a tail to strike the ground in order to impart a lever action on the body, lifting it into the air.

1.3.6 Static Model

have developed a robotic in isolation, then measured the dynamics in order to apply it to a function for a mobile robot.

1.4 Bio-Inspiration

1.5 Conclusion and Discussion

References

[1] G. Wenger, A. De, and D. E. Koditschek, "Frontal plane stabilization and hopping with a 2DOF tail," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 567–573. DOI: 10.1109/IROS.2016.7759110.

[2] T. Libby, T. Y. Moore, E. Chang-Siu, *et al.*, "Tail-assisted pitch control in lizards, robots and dinosaurs," *Nature*, vol. 481, no. 7380, pp. 181–184, 2012. DOI: doi:10.1038/nature10710.

[3] Z. Jianguo, Z. Tianyu, X. Ning, F. J. Cintrón, M. W. Mutka, and X. Li, "Controlling aerial maneuvering of a miniature jumping robot using its tail," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE, 2013, pp. 3802–3807. DOI: 10.1109/IROS.2013.6696900.

[4] R. Briggs, J.-W. Lee, M. Haberland, and S.-B. Kim, "Tails in biomimetic design: Analysis, simulation, and experiment," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 1473–1480. DOI: 10.1109/IROS.2012.6386240.

[5] O. V. Borisova, I. I. Borisov, and S. A. Kolyubin, "Analysis and modeling of galloping robot with actuated tail for balance," in *2020 International Conference Nonlinearity, Information and Robotics (NIR)*, IEEE, 2020, pp. 1–6.

[6] Z. Xiuli, G. Jiaqing, and Y. Yanan, "Effects of head and tail as swinging appendages on the dynamic walking performance of a quadruped robot," *Robotica*, vol. 34, no. 12, pp. 2878–2891, 2016. DOI: 10.1017/S0263574716000011.

[7] R. Sato, S. Hashimoto, A. Ming, and M. Shimojo, "Development of a flexible tail for legged robot," in *Mechatronics and Automation (ICMA), 2016 IEEE International Conference on*, IEEE, 2016, pp. 683–688. DOI: 10.1109/ICMA.2016.7558645.

[8] B. Simon, R. Sato, J.-Y. Choley, and A. Ming, "Development of a bio-inspired flexible tail system," in *2018 12th France-Japan and 10th Europe-Asia Congress on Mechatronics*, IEEE, 2018, pp. 230–235.

[9] A. L. Brill, A. De, A. M. Johnson, and D. E. Koditschek, "Tail-assisted rigid and compliant legged leaping," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, IEEE, 2015, pp. 6304–6311. DOI: 10.1109/IROS.2015.7354277.

Chapter 2

Creating a Configurable Payload for Instability Experiments

2.1 Introduction

In order to generate a diverse set of test data for the experiments in chapter **??**, a configurable payload was conceived, an object that could be configured to have a wide range of masses and COM. A series of test points can then be generated which have a specific mass and COM, and a matching algorithm can be used to find the configuration that mostly closely matches these parameters. The experiments in chapter **??** can then be run with each of these test points to generate the test data.

2.2 Payload Design

The payload consists of a matrix of cubes of various materials packed tightly into a Three Dimensional (3D) printed container. The cubes are designed to be changed after each experimental run to alter the mass and COM of the payload. A lid on the container prevents the cubes from falling out during the experiment, and the exterior design of the box may accommodate additional features to improving the handling of the payload by the robot arm.

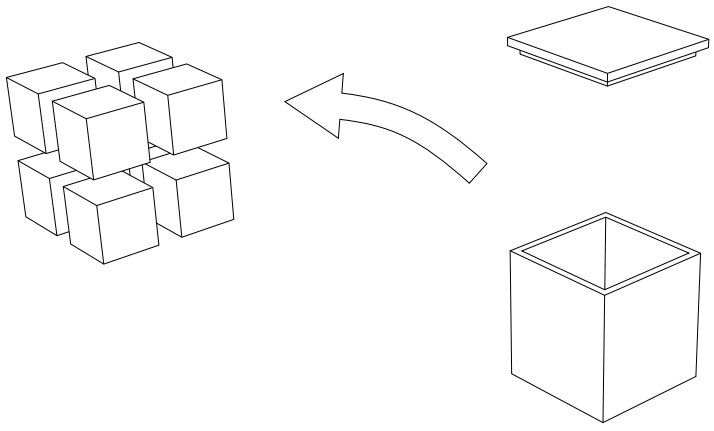


Figure 2.1: Concept drawing of the configurable payload.

Material	Variant	Density (kg m^{-3})
Wood	Pine	860
Plastic	Acrylic	1200
Aluminium	6082	2700
Steel	EN3B	8060

Table 2.1: The materials chosen for the cubes.

2.3 Payload Configuration Space

In order to find a configuration that closely matches a desired test point, first the payload has to be abstracted mathematically, so the mass and COM can be calculated for a given configuration. Firstly we can consider a positive real set of material densities $\mathcal{P} \in \mathbb{R}^+$, each element the density (in kg m^{-3}) of a material to be used:

$$\mathcal{P} = \{\rho_1, \rho_2 \dots \rho_n \mid \rho_i > 0\} \tag{2.1}$$

Each configuration can then be defined as an $n \times n \times n$ matrix $\boldsymbol{C}$, such that each element is an element of $\mathcal{P}$, where n^3 is the number of cubes in the matrix:

$$\boldsymbol{C} = (c_{ijk}) \in \mathbb{R}^{n^n} \mid (c_{ijk}) \in \mathcal{P} \tag{2.2}$$

2.3.1 Mass and COM functions

To calculate the mass of the configuration, we can take the sum of all the cube densities multiplied by their volume a^3, where a is the cube edge length, plus the container mass m_c:

$$M(\boldsymbol{C}) = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} c_{ijk} a^3 \right) + m_c \tag{2.3}$$

To calculate the COM, we can take the sum of each cube mass multiplied by its position relative to the centroid of the center cube (c_{222}), which can be calculated from the cube indexes ijk, plus the container COM $\boldsymbol{r}_c$ if non-zero:

$$R(\boldsymbol{C}) = \frac{\left(\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} c_{ijk} a^4 \left(\begin{bmatrix} i & j & k \end{bmatrix} - n + 1 \right) \right) + \boldsymbol{r}_c}{M(\boldsymbol{C})} \tag{2.4}$$

2.4 Test Points

With $\mathcal{Z}$ as the set of all permutations of $\boldsymbol{C}$, test points can be derived from subsets of $\mathcal{Z}$ defined either by logical expressions on $\mathcal{Z}$, or the nearest neighbours of $\mathcal{Z}$ from a set of co-

ordinates. A test point is defined as a $\mathbb{R}^4$ vector containing the target mass and COM concatenated as $\begin{bmatrix} m_i & \boldsymbol{r}_i \end{bmatrix}$.

2.4.1 Extrema Set ($\mathcal{E}$)

The extrema set is designed to test the extremas of the space of $\mathcal{Z}$ for both $M(\mathcal{Z})$ and $R(\mathcal{Z})$. The extrema set if defined from a set of logical constraints. The first two constraints of the set find the maximum and minimum values of the payload mass using $M(\mathcal{Z})$, and the next four constraints use the payload COM using $M(\mathcal{C})$ to get the maximum and minimum values of the x and y component of the COM. Finally, the last four constraints define the diagonal maximum and minimum values where the COM components match $x = y$ or $x = -y$.

$$\mathcal{E} = \left\{ \boldsymbol{x} \subset \mathcal{Z} \mid \left\{ \begin{array}{l} M(\boldsymbol{x}) = \max\{M(\mathcal{Z})\} \\ M(\boldsymbol{x}) = \min\{M(\mathcal{Z})\} \\ R(\boldsymbol{x})_x = \max\{R(\mathcal{Z})_x\} \\ R(\boldsymbol{x})_x = \min\{R(\mathcal{Z})_x\} \\ R(\boldsymbol{x})_y = \max\{R(\mathcal{Z})_y\} \\ R(\boldsymbol{x})_y = \min\{R(\mathcal{Z})_y\} \\ R(\boldsymbol{x})_x = \max\{R(\mathcal{Z})_x\} \wedge R(\boldsymbol{x})_x = R(\boldsymbol{x})_y \\ R(\boldsymbol{x})_x = \min\{R(\mathcal{Z})_x\} \wedge R(\boldsymbol{x})_x = R(\boldsymbol{x})_y \\ R(\boldsymbol{x})_x = \max\{R(\mathcal{Z})_x\} \wedge R(\boldsymbol{x})_x = -R(\boldsymbol{x})_y \\ R(\boldsymbol{x})_x = \min\{R(\mathcal{Z})_x\} \wedge R(\boldsymbol{x})_x = -R(\boldsymbol{x})_y \end{array} \right\} \right. \tag{2.5}$$

Mass Limited $\mathcal{E}$

When this set was generated, it was found that $M(\mathcal{E}_1)$ was greater than the chosen robot arm could safely lift. Therefore, $M(\boldsymbol{x}) = \max\{M(\mathcal{Z})\}$ in $\mathcal{E}$ was changed to $\begin{bmatrix} m_{max} & 0 & 0 & 0 \end{bmatrix}$ where m_{max} is the safe mass limit that the robot arm can lift. One of the search methods described in section 2.5 can then be used to find the nearest point in configuration space.

2.4.2 Cube Set ($\mathcal{C}$)

The cube set is defined by the vertices of a cube of size b centred around the COM origin $\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$.

$$\mathcal{C} = \left\{ \boldsymbol{x} \subset \mathcal{C} \mid \begin{bmatrix} \pm\frac{b}{2} & \pm\frac{b}{2} & \pm\frac{b}{2} \end{bmatrix} = \boldsymbol{x} \right\} \tag{2.6}$$

2.4.3 Balanced Set ($\mathcal{B}$)

The balanced set is defined by q points in $\mathcal{C}$ subject to the constraint $R(\boldsymbol{x})_x = 0 \wedge R(\boldsymbol{x})_y = 0$. This can be defined as a "balanced" set as the COM x and y components are both zero. The points are evenly spaced between the maximum and minimum mass as defined in section 2.4.1.

$$\begin{aligned} m_r &= \frac{\max\{M(\mathcal{Z})\} - \min\{M(\mathcal{Z})\}}{q+1} \\ \boldsymbol{z} &= \begin{bmatrix} m_r & 2m_r & \cdots & qm_r \end{bmatrix} \\ \mathcal{B} &= \{\boldsymbol{x} \subset \mathcal{C} \mid z_i = \boldsymbol{x} \wedge R(\boldsymbol{x})_x = 0 \wedge R(\boldsymbol{x})_y = 0\} \end{aligned} \tag{2.7}$$

2.5 Test Point Matching Search Methods

While we are guaranteed an exact result for elements of $\mathcal{E}$ as every element is defined by constraints on known configurations, for $\mathcal{C}$ and $\mathcal{B}$ as the elements are defined numerically, there is no guarantee that any element will have an exact match in solution space. The cardinality of $\mathcal{Z}$ is also important. It is defined as $|\mathcal{Z}| = |\mathcal{P}|^{n^n}$ which increases super exponentially with n. For example, when $|\mathcal{P}| = 4$, $n = 2$ results in 256 permutations and $n = 3$ results in approximately 1.8×10^{16} permutations. It's very clear that when $n > 2$ for non-trivial cardinalities of $\mathcal{P}$, any kind of brute-force method is not computationally tractable. The only exception is for $\mathcal{E}_1$ and $\mathcal{E}_2$, where the solution is trivial as $\mathcal{E}_1 = \boldsymbol{C} \mid \min(\mathcal{P}) \forall c_{ijk}$ and $\mathcal{E}_2 = \boldsymbol{C} \mid \max(\mathcal{P}) \forall c_{ijk}$. Therefore, we investigated both a brute-force nearest neighbour method for when $n = 2$, and a simulated annealing method for when $n = 3$.

2.5.1 Simulated Annealing ($n = 3$)

Simulated annealing [1] is a modification to a gradient descent optimisation that allows the algorithm the chance to "jump out" of local minima early on (even though the approximation becomes temporarily worse). However, as the number of remaining steps decreases, that probability becomes smaller, becoming more and more like gradient descent. First, like any gradient descent algorithm, two things need to be generated, the initial configuration $\boldsymbol{C_0}$, which can be random or manually selected, and the function $\mathcal{N}(\boldsymbol{C})$ which creates a set of all the "neighbours" of $\boldsymbol{C}$. In this case, this can be defined as the subset of $\mathcal{Z}$ where the difference between $\boldsymbol{C}$ and an element of $\mathcal{N}(\boldsymbol{C})$ is one and only one $c_{ijk} \neq c_{ijk}$:

$$\mathcal{N}(\boldsymbol{C}) = \{x \subset \mathcal{Z} \mid \exists!\,(x_{ijk} \neq c_{ijk})\} \tag{2.8}$$

Then the simulated annealing function can be described as follows:

1. Set $\boldsymbol{C}$ to the initial permutation $\boldsymbol{C_0}$.

2. For each of the optimisation steps:

   (a) Set the temperature value t with function $T\left(\frac{k_{max}}{k}\right)$ which takes into account the number of remaining steps.

   (b) Set $\boldsymbol{C}_{new}$ as a random element from the set of all neighbours of $\boldsymbol{C}$ as defined by $\mathcal{N}(\boldsymbol{C})$.

   (c) Use acceptance probability function $P\left(E\left(\boldsymbol{C}\right), E\left(\boldsymbol{C}_{new}\right), t\right)$ where $E\left(\boldsymbol{C}\right)$ is the energy function (in this case the absolute difference $E\left(\boldsymbol{C}, g\right) = \left|M\left(\boldsymbol{C}\right) - g\right|$ or $E\left(\boldsymbol{C}, g\right) = \left|R\left(\boldsymbol{C}\right) - g\right|$ where g is the target could be used to look for a lower energy state in either mass or COM) to generate a value. Note this function is dependent on the temperature t.

   (d) Compare that value with a random uniformly distributed real number between 0 and 1. If greater than or equal to, then replace $\boldsymbol{C}$ with $\boldsymbol{C}_{new}$. Otherwise, keep it the same.

   (e) Repeat with $\boldsymbol{C}$ until there are no remaining steps.

3. Return the approximated permutation $\boldsymbol{C}$.

$\boldsymbol{C} = \boldsymbol{C}_0$
for $k \leftarrow 1, k_{max}$ **do**
  $t = T\left(\frac{k_{max}}{k}\right)$
  $\boldsymbol{C}_{new} = \mathcal{N}\left(\boldsymbol{C}\right) \xleftarrow{R} x$
  if $P\left(E\left(\boldsymbol{C}\right), E\left(\boldsymbol{C}_{new}\right), t\right) \geq x \sim U\left([0, 1]\right)$ **then**
    $\boldsymbol{C} = \boldsymbol{C}_{new}$
  end if
end for
return $\boldsymbol{C}$

Using this algorithm an array of approximate configurations can be easily generated from a desired array of test points. Simulated annealing can also be adapted for multi-objective optimisation [2], so it is possible to generate test points that approximate a desired mass and COM simultaneously.

Cooling Function

The function which controls the probability of exiting local minima (known as the *temperature*) is known as the *cooling function*. This function can be any function which monotonically decreases (except in adaptive simulated annealing where it is dependant on the accuracy of the current approximation). Different functions will result in a different cooling profile, generally decreasing quickly in the first few steps, and then slowing down after that.

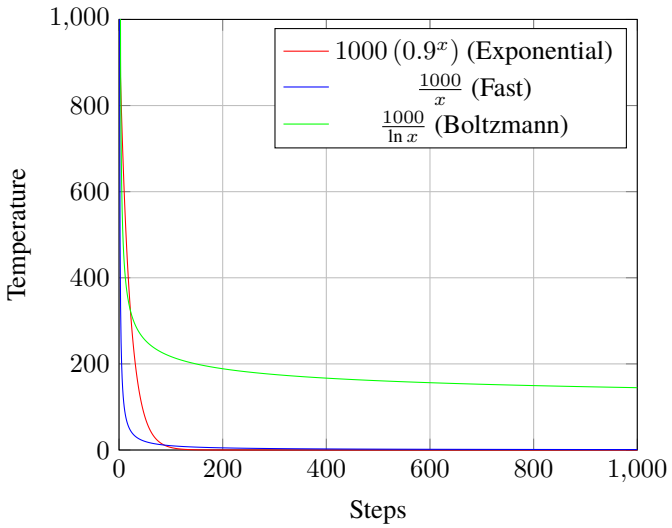


Figure 2.2: Various temperature cooling profiles for simulated annealing, assuming 1000 steps.

2.5.2 Brute-Force Nearest Neighbour ($n = 2$)

If $\mathcal{Z}$ is suitably small, as it is when $n = 2$ then a brute-force method can be used which is guaranteed to find the nearest element to the target within a finite time. This can be done by calculating the L2 norms between the target vector t and all the elements of $\mathcal{Z}$ and finding the minimum. If several elements of $\mathcal{Z}$ have the minimum norm, then one is chosen at random from this set.

$$NN\left(t, \mathcal{Z}\right) = \min\left\{\left\|t - x\right\|_2 \mid \forall x \in \mathcal{Z}\right\} \tag{2.9}$$

2.5.3 Selected Method

Initially an $n = 3$ configuration was used with the acceptance probability function *Rule M* from [2]. This is a weighted blend of two other algorithms defined in the paper, *Rule P* and *Rule W* with a weighting coefficient $\alpha \in (0, 1) \subset \mathbb{R}$. There is also a weighting vector for each element of the test point $\boldsymbol{w} \in \mathbb{R}^4 \mid w_i \in (0, 1)$.

$$P\left(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, t\right) = \alpha \underbrace{\prod_{i=1}^{m} \min\left\{1, e^{\frac{w_i(x_i - y_i)}{t}}\right\}}_{\text{Rule P}} + (1 - \alpha) \underbrace{\min\left\{1, \max_{i=1,\ldots,m}\left\{1, e^{\frac{w_i(x_i - y_i)}{t}}\right\}\right\}}_{\text{Rule W}} \tag{2.10}$$

Unfortunately it was difficult to find a stable and consistent result even after a long time running the algorithm.

Therefore as an alternative, the $n = 2$ configuration was used, with larger cubes to compensate. This successfully produced all three test point sets, after some small adjustments of b in order to get closer to a single configuration for each point of $\mathcal{C}$.

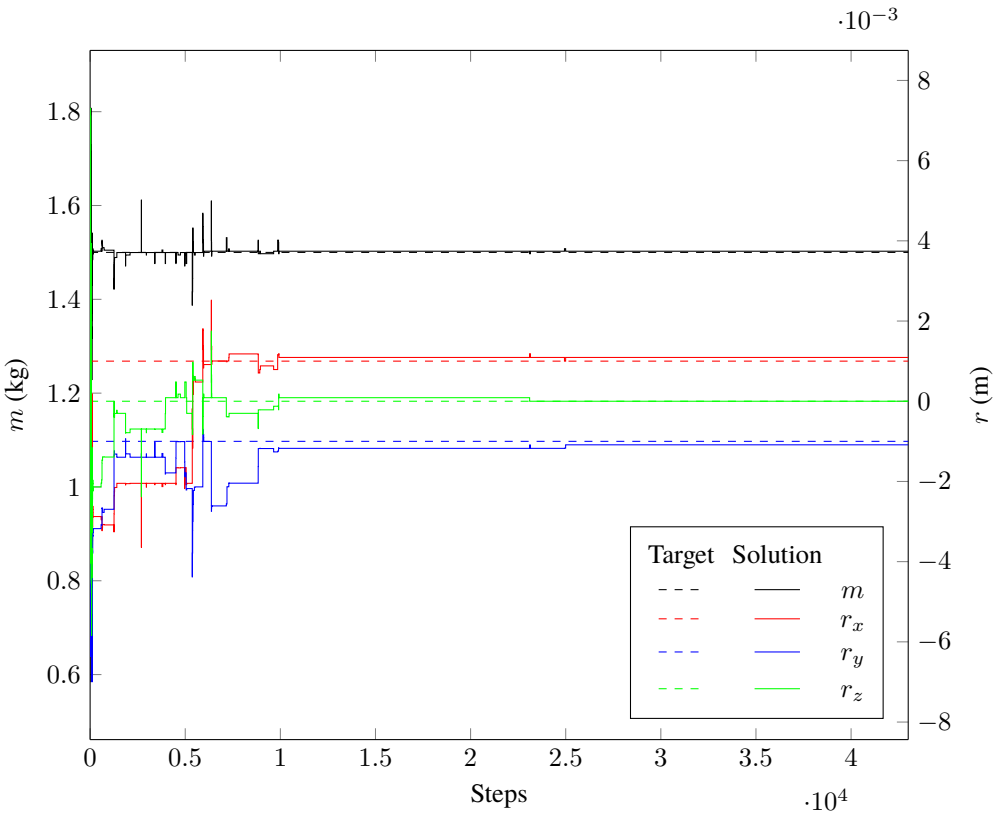


Figure 2.3: Simulated annealing output for the target $[1.5 \quad 0.001 \quad 0.001 \quad 0.001]$ with $\alpha = 0.997$ and even weighting $w_m, \boldsymbol{w_r} = 0.25$ for 4.3×10^4 steps.

m	$\boldsymbol{r}$
Extrema Set ($\mathcal{E}$)	
1.751	$[0.012 \quad 0.000 \quad 0.000]$
1.751	$[0.000 \quad 0.012 \quad 0.000]$
1.751	$[-0.012 \quad 0.000 \quad 0.000]$
1.751	$[-0.000 \quad -0.012 \quad 0.000]$
0.516	$[0.000 \quad 0.000 \quad 0.000]$
1.133	$[0.010 \quad 0.010 \quad 0.000]$
1.133	$[-0.010 \quad -0.010 \quad 0.000]$
1.133	$[0.010 \quad -0.010 \quad 0.000]$
1.133	$[-0.010 \quad 0.010 \quad 0.000]$

Table 2.2: Table of the vectors of $\mathcal{E}$, excluding the mass-limited element.

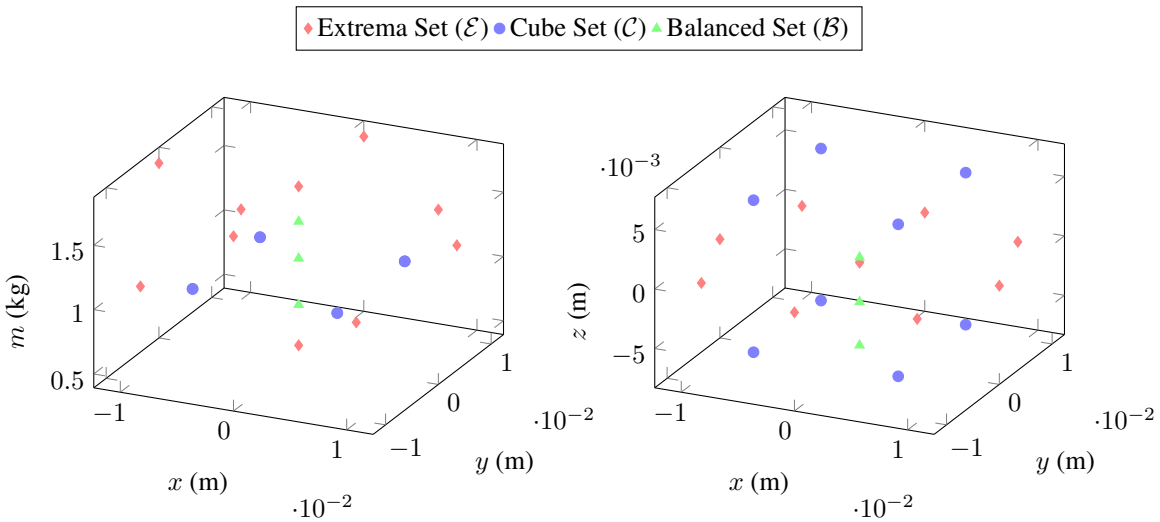


Figure 2.4: Mass and COM coordinates for each test point set.

Target		Nearest		L2 Norm Error
m	$\boldsymbol{r}$	m	$\boldsymbol{r}$	
Extrema Set ($\mathcal{E}$)				
1.770	$\begin{bmatrix}0.000 & 0.000 & 0.000\end{bmatrix}$	1.751	$\begin{bmatrix}-0.000 & -0.000 & -0.000\end{bmatrix}$	1.922×10^{-2}
Cube Set ($\mathcal{C}$)				
$*$	$\begin{bmatrix}-0.007 & -0.007 & -0.007\end{bmatrix}$	1.061	$\begin{bmatrix}-0.006 & -0.006 & -0.006\end{bmatrix}$	1.055×10^{-3}
$*$	$\begin{bmatrix}-0.007 & -0.007 & 0.007\end{bmatrix}$	1.061	$\begin{bmatrix}-0.006 & 0.006 & -0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}-0.007 & 0.007 & -0.007\end{bmatrix}$	1.061	$\begin{bmatrix}0.006 & -0.006 & -0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}-0.007 & 0.007 & 0.007\end{bmatrix}$	1.061	$\begin{bmatrix}0.006 & 0.006 & -0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}0.007 & -0.007 & -0.007\end{bmatrix}$	1.061	$\begin{bmatrix}-0.006 & -0.006 & 0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}0.007 & -0.007 & 0.007\end{bmatrix}$	1.061	$\begin{bmatrix}-0.006 & 0.006 & 0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}0.007 & 0.007 & -0.007\end{bmatrix}$	1.061	$\begin{bmatrix}0.006 & -0.006 & 0.006\end{bmatrix}$	1.895×10^{-2}
$*$	$\begin{bmatrix}0.007 & 0.007 & 0.007\end{bmatrix}$	1.061	$\begin{bmatrix}0.006 & 0.006 & 0.006\end{bmatrix}$	1.055×10^{-3}
Balanced Set ($\mathcal{B}$)				
0.516	$\begin{bmatrix}0.000 & 0.000 & *\end{bmatrix}$	0.832	$\begin{bmatrix}0.000 & 0.000 & -0.003\end{bmatrix}$	3.156×10^{-1}
1.258	$\begin{bmatrix}0.000 & 0.000 & *\end{bmatrix}$	1.192	$\begin{bmatrix}-0.000 & -0.000 & 0.000\end{bmatrix}$	6.630×10^{-2}
2.000	$\begin{bmatrix}0.000 & 0.000 & *\end{bmatrix}$	1.478	$\begin{bmatrix}-0.000 & -0.000 & -0.007\end{bmatrix}$	5.219×10^{-1}

Table 2.3: Table of the target and actual vectors for $\mathcal{C}$, $\mathcal{B}$ and the mass limited element of $\mathcal{E}$ with the L2 norm error. $*$ notation indicates "don't care" and is excluded from the search algorithm.

2.6 Conclusion and Discussion

References

[1] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[2] P. Serafini, "Simulated annealing for multi objective optimization problems," in *Multiple criteria decision making*, Springer, 1994, pp. 283–292.

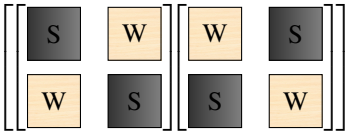
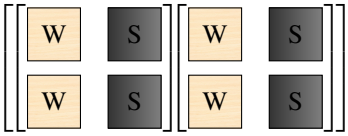
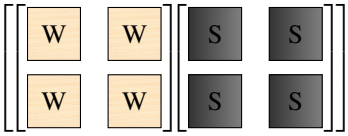
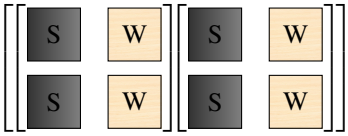
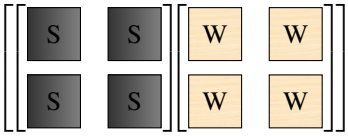
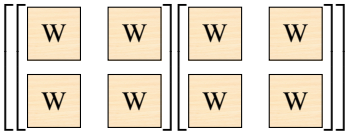
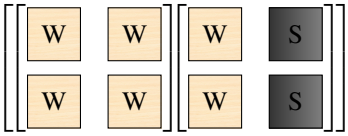
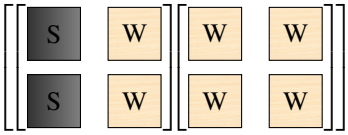
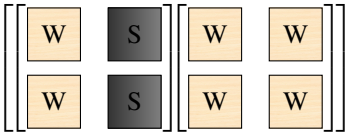
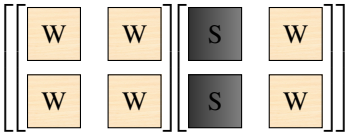
m	$\boldsymbol{r}$	Material Matrix	3D Preview
		Extrema Set ($\mathcal{E}$)	
1.751	$[-0.000 \quad -0.000 \quad -0.000]$		
1.751	$[0.012 \quad 0.000 \quad 0.000]$		
1.751	$[0.000 \quad 0.012 \quad 0.000]$		
1.751	$[-0.012 \quad 0.000 \quad 0.000]$		
1.751	$[-0.000 \quad -0.012 \quad 0.000]$		
0.516	$[0.000 \quad 0.000 \quad 0.000]$		
1.133	$[0.010 \quad 0.010 \quad 0.000]$		
1.133	$[-0.010 \quad -0.010 \quad 0.000]$		
1.133	$[0.010 \quad -0.010 \quad 0.000]$		
1.133	$[-0.010 \quad 0.010 \quad 0.000]$		

Table 2.4

m	$\boldsymbol{r}$	Material Matrix	3D Preview
		Cube Set ($\mathcal{C}$)	
1.061	$[-0.006 \quad -0.006 \quad -0.006]$	[[[S, A], [A, W]], [[A, W], [W, W]]]	
1.061	$[-0.006 \quad 0.006 \quad -0.006]$	[[[A, W], [W, W]], [[S, A], [A, W]]]	
1.061	$[0.006 \quad -0.006 \quad -0.006]$	[[[A, S], [W, A]], [[W, A], [W, W]]]	
1.061	$[0.006 \quad 0.006 \quad -0.006]$	[[[W, A], [W, W]], [[A, S], [W, A]]]	
1.061	$[-0.006 \quad -0.006 \quad 0.006]$	[[[A, W], [S, A]], [[W, W], [A, W]]]	
1.061	$[-0.006 \quad 0.006 \quad 0.006]$	[[[W, W], [A, W]], [[A, W], [S, A]]]	
1.061	$[0.006 \quad -0.006 \quad 0.006]$	[[[W, A], [A, S]], [[W, W], [W, A]]]	
1.061	$[0.006 \quad 0.006 \quad 0.006]$	[[[W, W], [W, A]], [[W, A], [A, S]]]	

Table 2.5

m	$\boldsymbol{r}$	Material Matrix	3D Preview
		Balanced Set ($\mathcal{B}$)	
0.832	$[0.000 \quad 0.000 \quad -0.003]$	[[[A, A], [W, W]], [[W, A], [A, W]]]	
1.192	$[-0.000 \quad -0.000 \quad 0.000]$	[[[P, W], [S, P]], [[W, S], [P, P]]]	
1.478	$[-0.000 \quad -0.000 \quad -0.007]$	[[[S, A], [P, A]], [[A, S], [A, P]]]	

Table 2.6

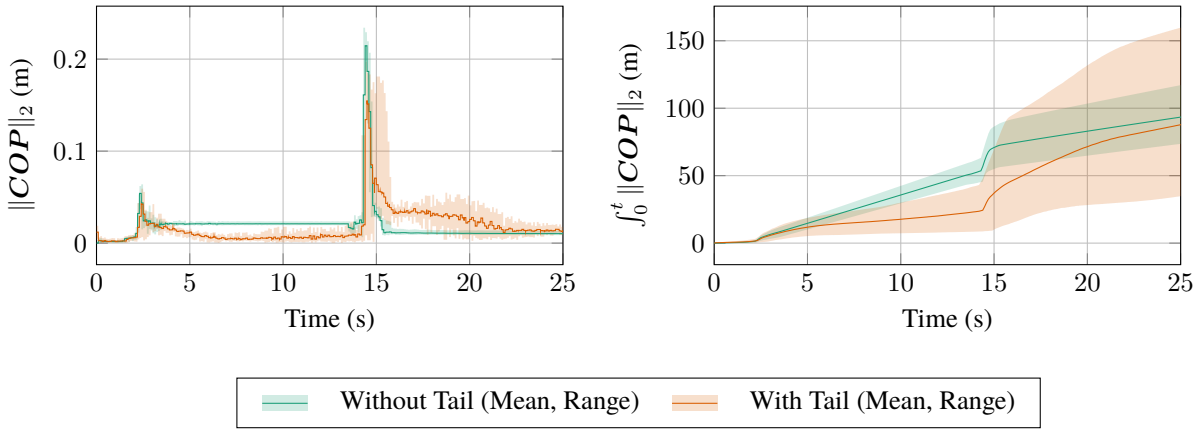
Appendices

Appendix A

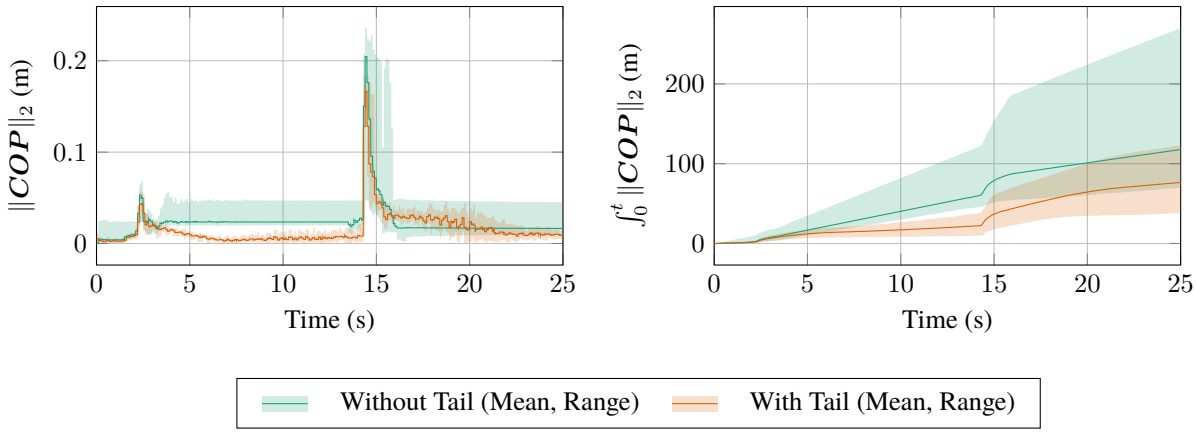
Graphs of the Test Rig Data

A.1 Extrema Set ($\mathcal{E}$)

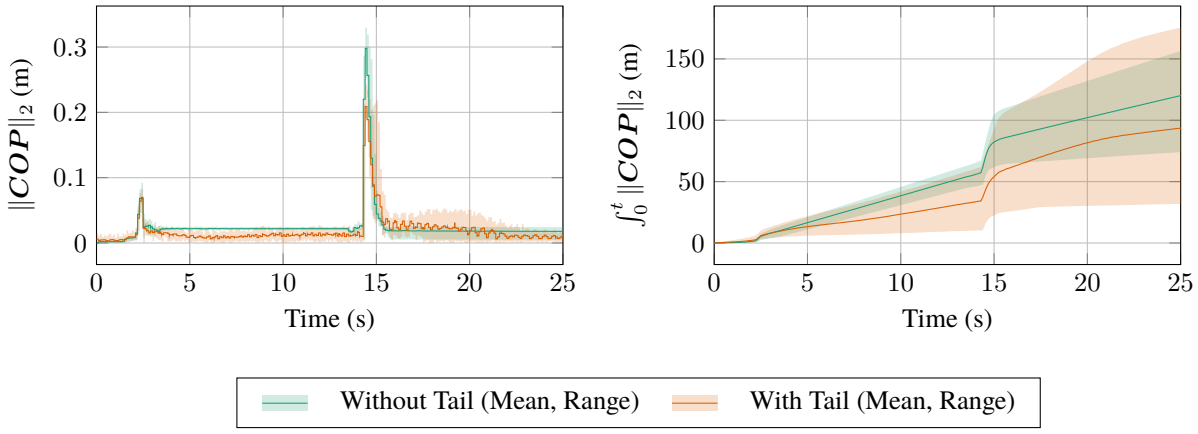
$$1.291 \begin{bmatrix} -0.000 & 0.000 & -0.008 \end{bmatrix}$$



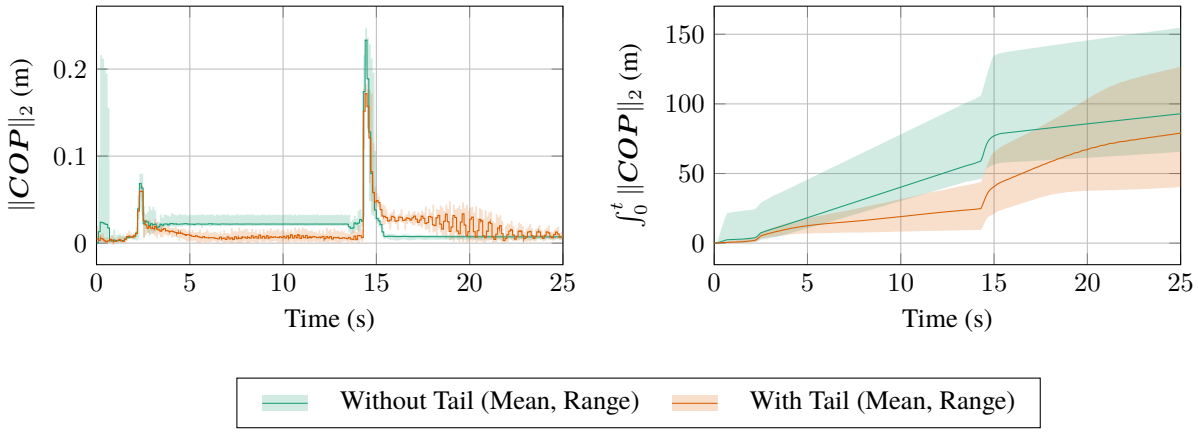
$$1.291 \begin{bmatrix} 0.011 & -0.000 & 0.000 \end{bmatrix}$$



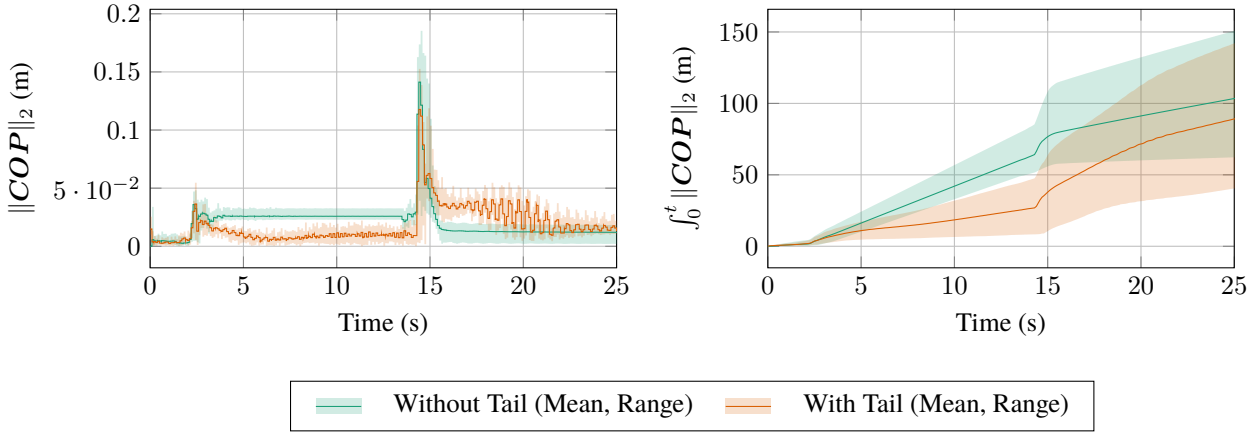
$1.291\,[0.000 \quad 0.011 \quad 0.000]$



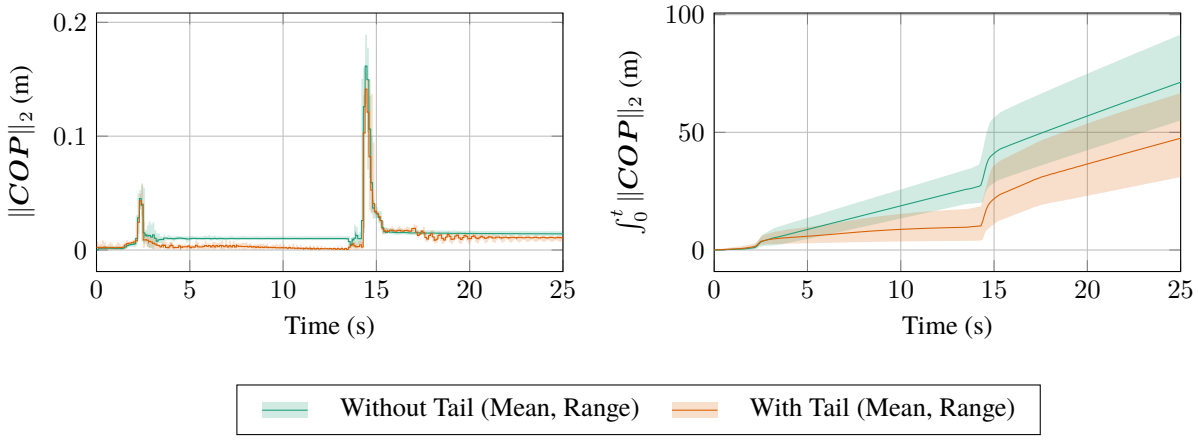
$1.291\,[-0.011 \quad -0.000 \quad -0.000]$



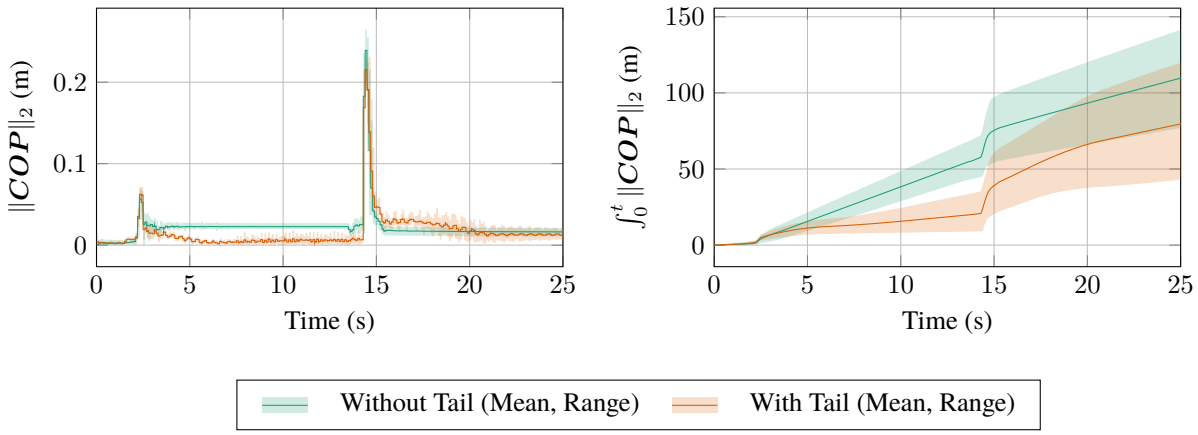
$1.291\,[-0.000 \quad -0.011 \quad -0.000]$



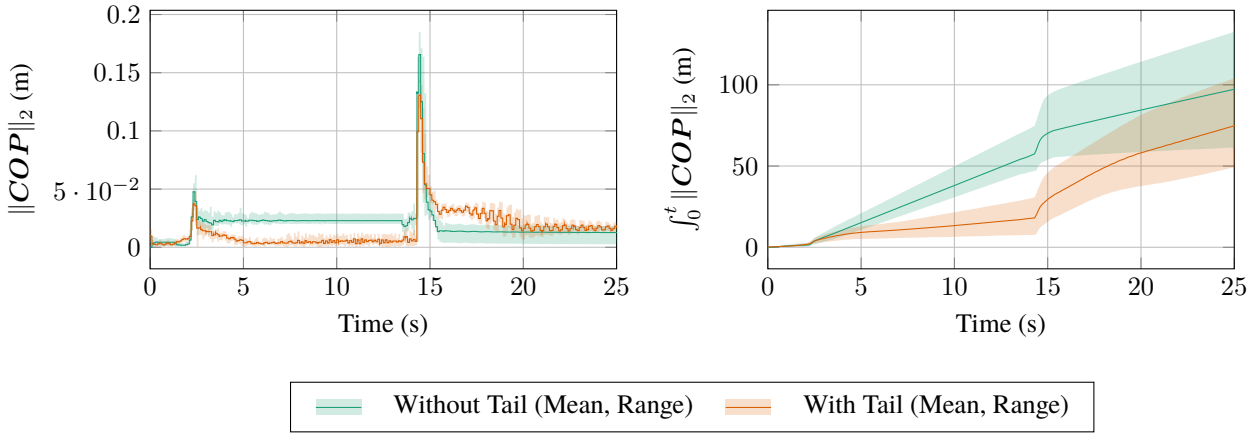
$0.516\ [0.000 \quad 0.000 \quad 0.000]$



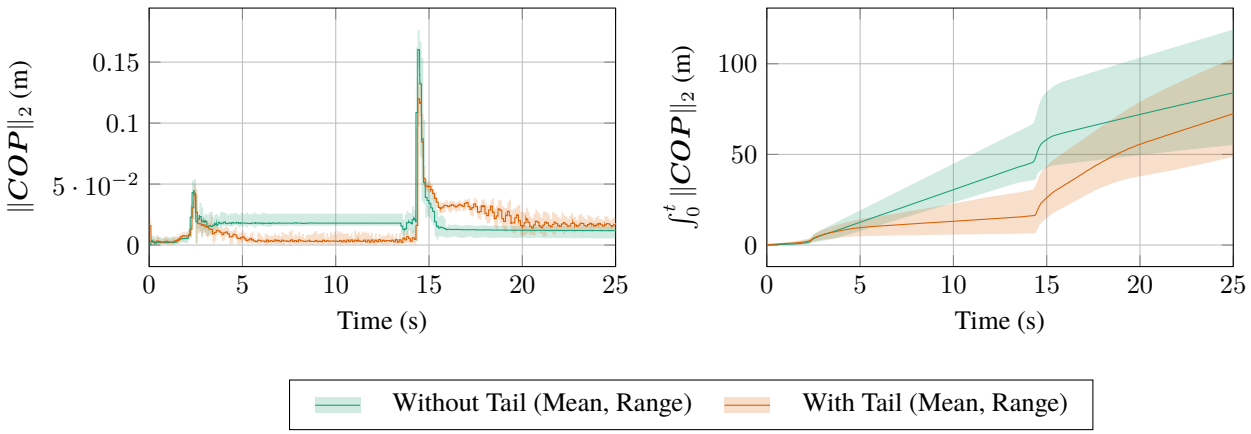
$1.133\ [0.010 \quad 0.010 \quad 0.000]$



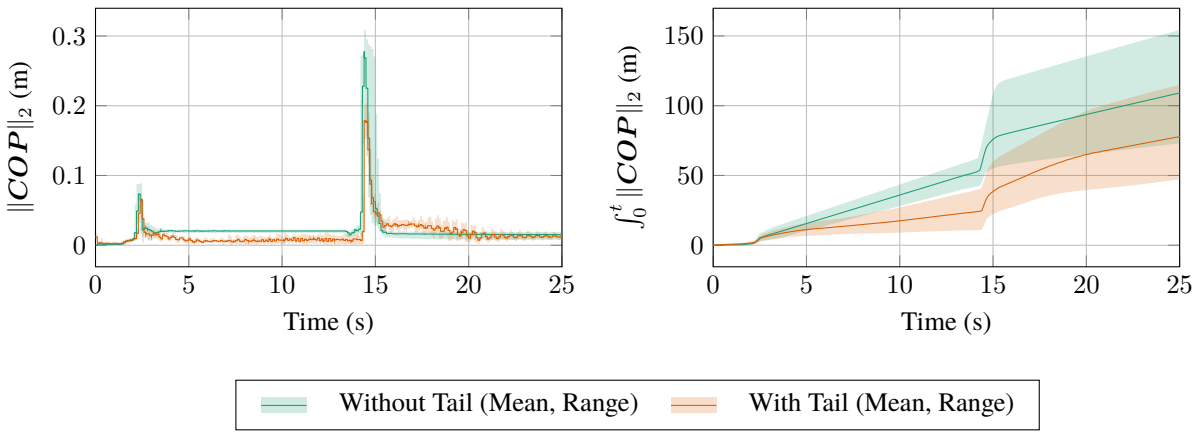
$1.133\ [-0.010 \quad -0.010 \quad 0.000]$



$1.133\,[0.010 \quad -0.010 \quad 0.000]$

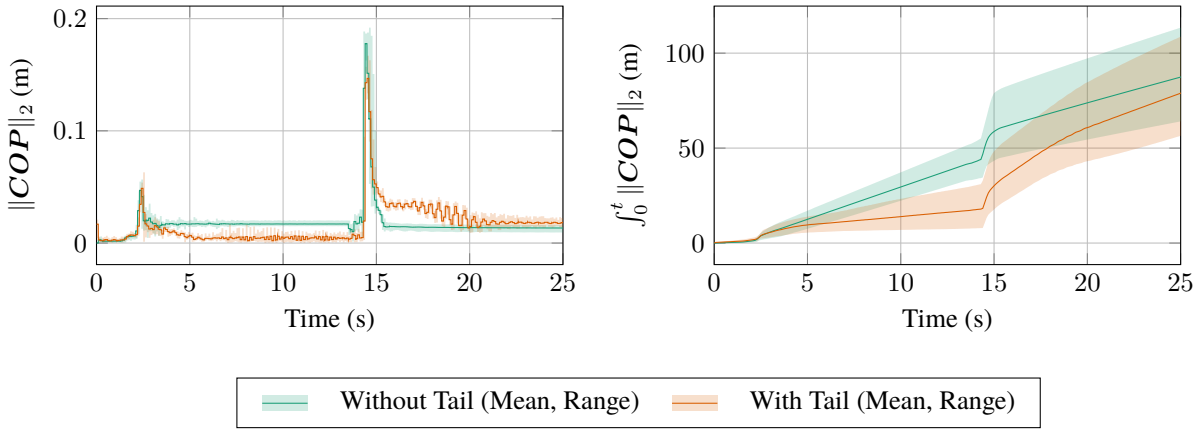


$1.133\,[-0.010 \quad 0.010 \quad 0.000]$

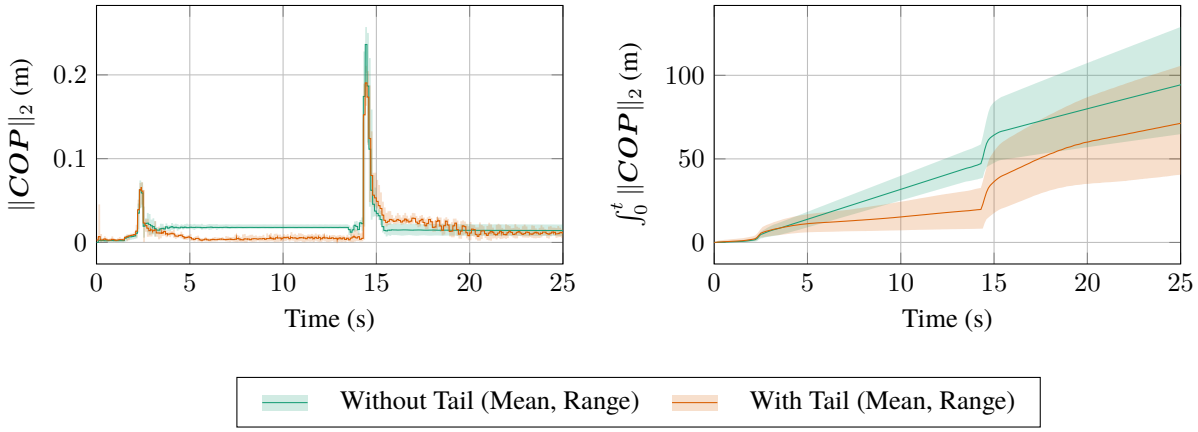


A.2 Cube Set ($\mathcal{C}$)

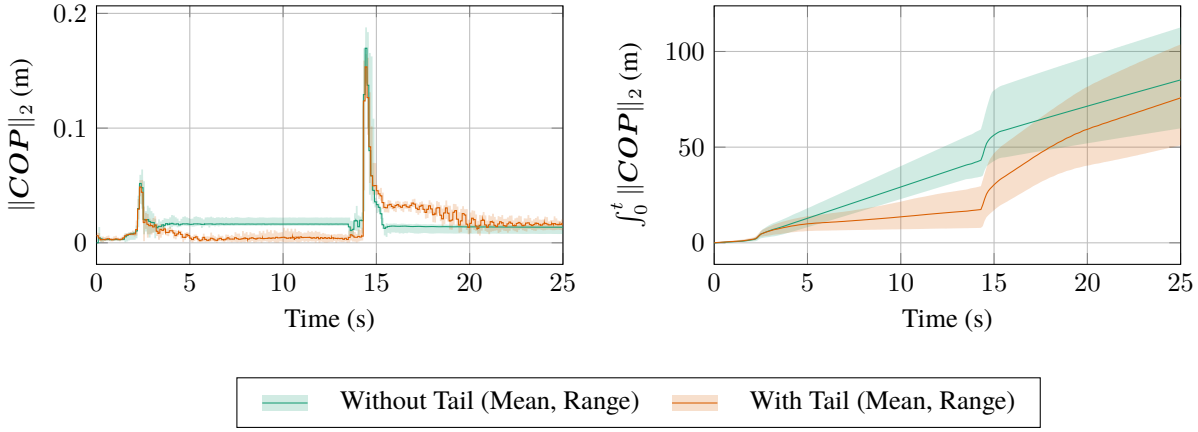
$1.061\,[-0.006 \quad -0.006 \quad -0.006]$



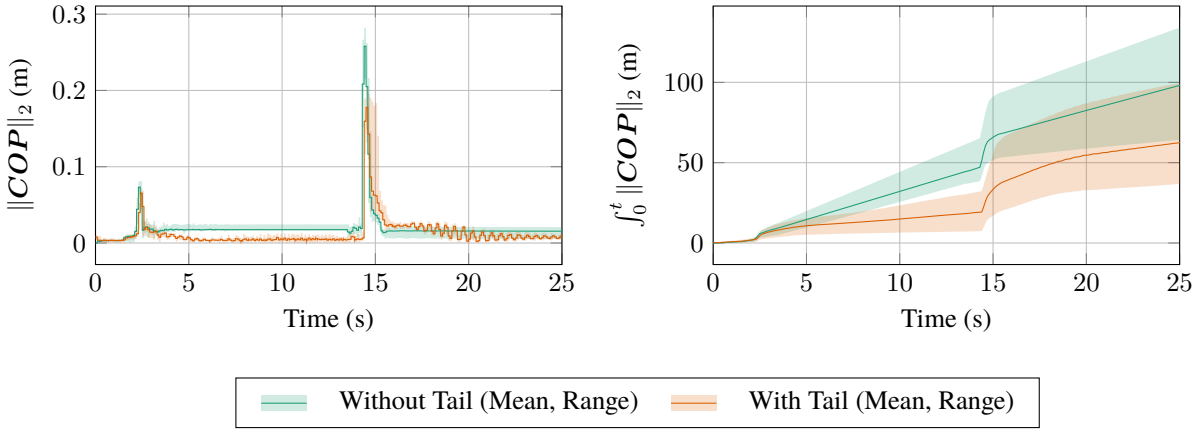
$1.061\,[-0.006 \quad 0.006 \quad -0.006]$



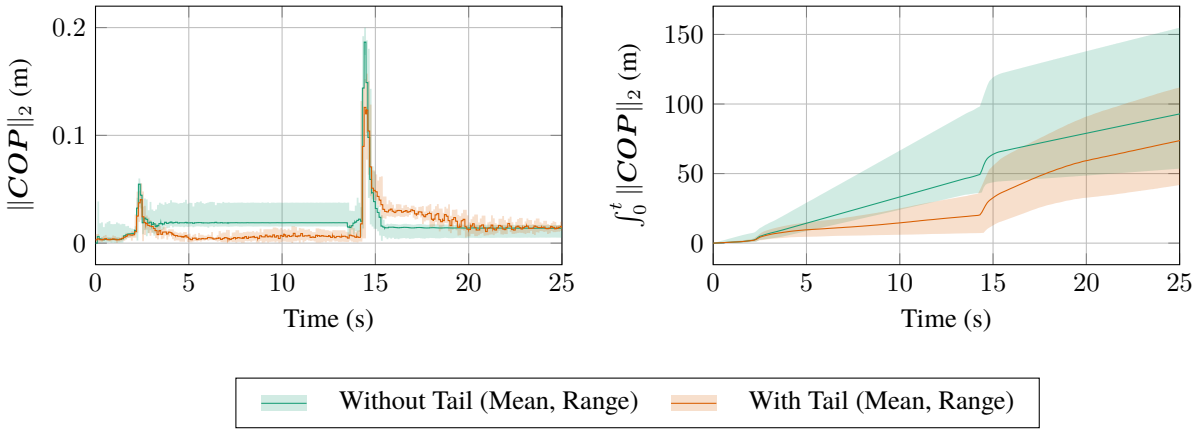
$1.061\,[0.006 \quad -0.006 \quad -0.006]$



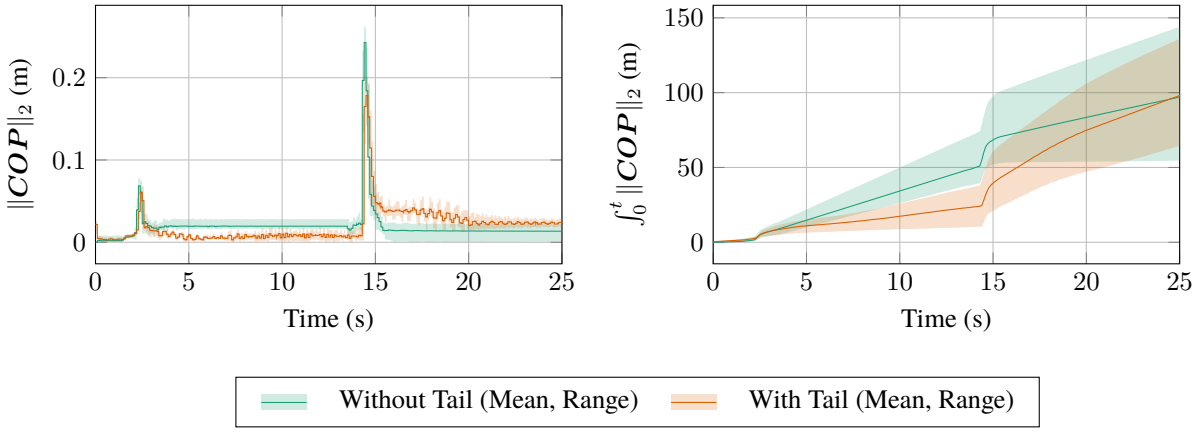
$1.061\,[0.006 \quad 0.006 \quad -0.006]$



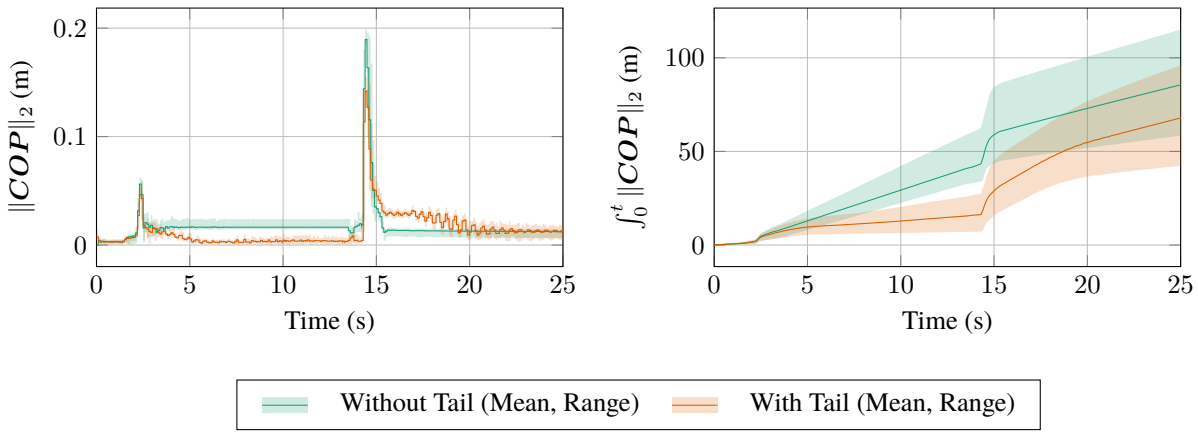
$1.061\,[-0.006 \quad -0.006 \quad 0.006]$



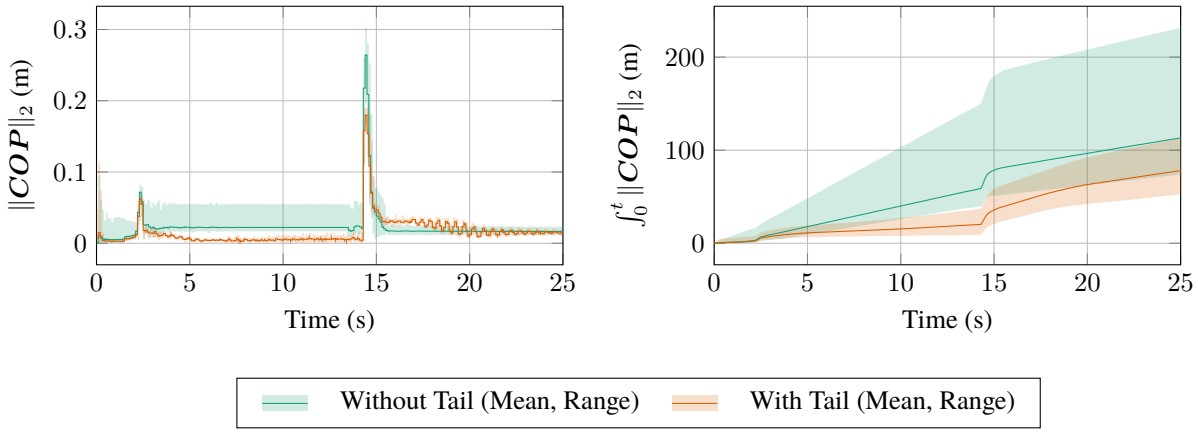
$1.061\,[-0.006 \quad 0.006 \quad 0.006]$



$1.061\,[0.006 \quad -0.006 \quad 0.006]$

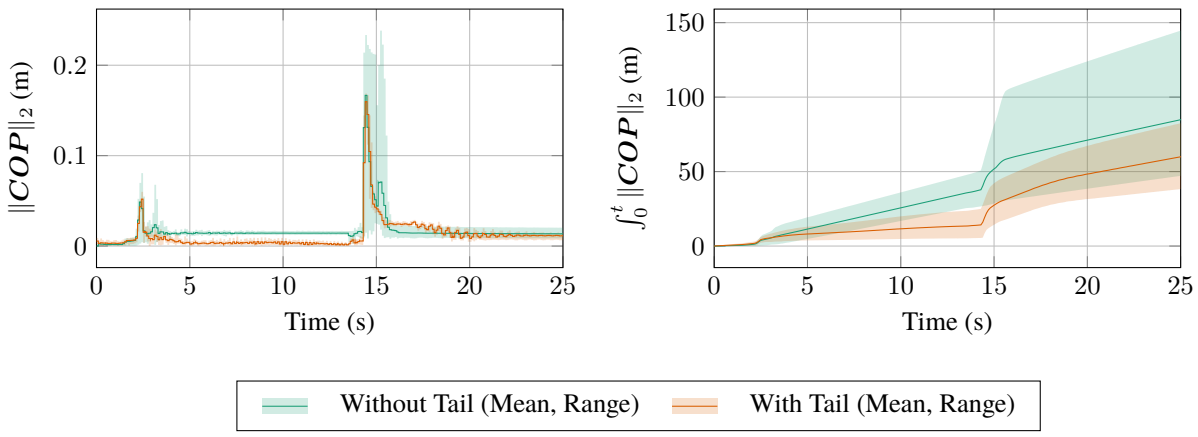


$1.061\,[0.006 \quad 0.006 \quad 0.006]$

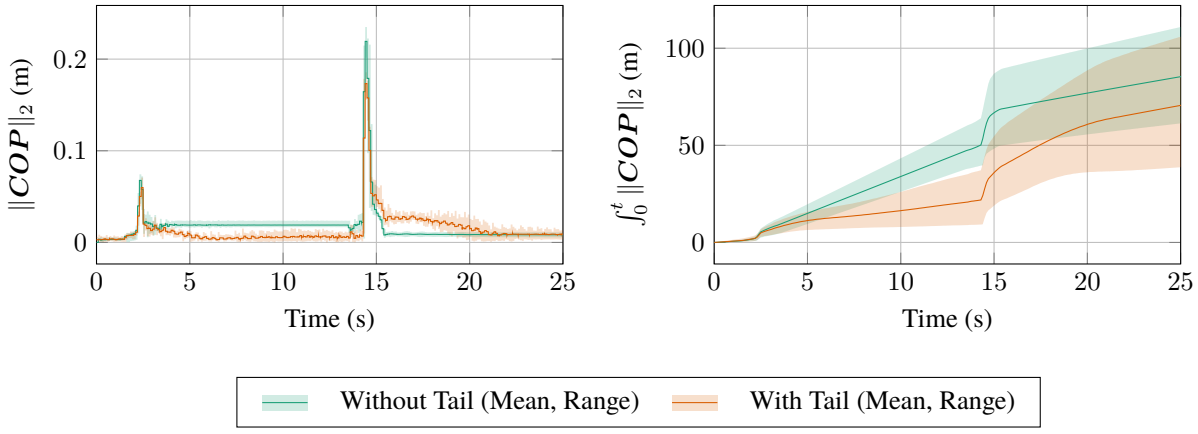


A.3 Balanced Set ($\mathcal{B}$)

$$0.832\,[0.000 \quad 0.000 \quad -0.003]$$



$$1.192\,[-0.000 \quad -0.000 \quad 0.000]$$



$$1.291\,[-0.000 \quad 0.000 \quad -0.008]$$