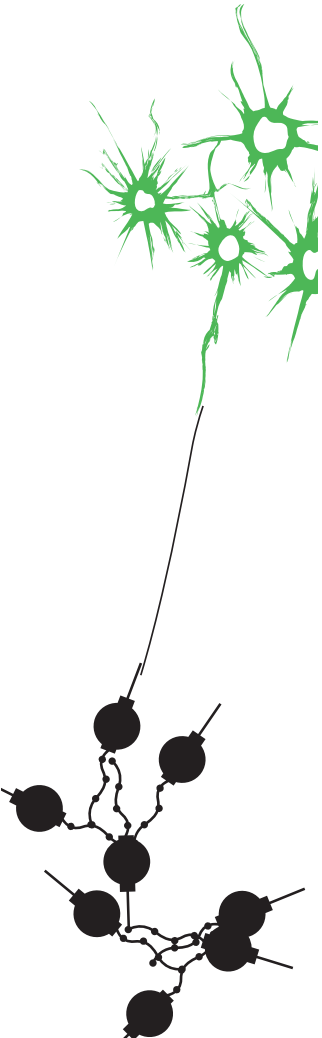


RAM ROBOTICS AND MECHATRONICS

IMPACT AWARE AERIAL ROBOTICS

G. (Gayatri) Indukumar

MSC ASSIGNMENT

Committee:
dr. ir. D. Dresscher
C. Gabellieri, Ph.D
prof. dr. ir. A. Franchi
dr. A. Saccon
dr. T.H.F. Hartman

September, 2023

051RaM2023
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Abstract

These days robots are increasingly interacting with their environment. However, there is still a lot of room for improvement in improving the efficiency of interaction tasks. Most applications perform contact tasks at near-zero velocities in order to be able to preserve the robot's stability. The dynamic contact transition problem studies contact tasks that take advantage of impacts and establish contacts at relatively higher velocities. It can increase task execution efficiency, reduce net energy consumption and increase the repertoire of tasks a robot is capable of performing. This problem has been effectively applied to manipulators and humanoids. This project explores the possibility of extending dynamic contact transition to aerial robots.

Aerial robots are a growing presence in the robotics industry owing to their unique vantage offering access to remote and dangerous environments. Most aerial robotics systems deployed today suffer from short flight times owing to limited power capabilities. Additionally, interaction with the environment is complicated by the problems encountered when performing tasks from a floating base. Overcoming these difficulties usually results in compliant control algorithms for lower impact velocities or complicated hardware solutions that increase the cost incurred and energy used. Successfully implementing dynamic contact transition in aerial robots offers a possibility to address some of these issues by enabling simpler systems like quadrotors to perform interaction tasks like swooping, intercepting a target in mid-air or force application.

The dynamic contact problem is explored here through the use case of swooping motion, widely seen in birds. To develop a test bed simulation scenario, key features of the motion are identified and translated into a trajectory-tracking task for the quadrotor. Three different scenarios of impact events are modelled based on the design of the object being impacted and the task executed.

The proposed controller extends the range of desired velocities where the quadrotor performs the trajectory tracking task utilising the dynamic contact transition problem. It is inspired by the Interaction planning problem and makes use of robotic reflex reactions. The scope of the proposed solution is set to high-impact tasks at low altitudes on systems with input saturation. The hypothesis that altitude factors into the task's success is confirmed through some high-altitude simulation tests. Furthermore, an error space is introduced that distinguishes post-impact behaviour where the quadrotor successfully executes the dynamic contact transition task and instances where it fails. The proposed controller is observed to increase the region in error space where the quadrotor succeeds. Finally, exploring the impact-variant subspace of the problem highlights some challenges involved in adapting dynamic contact transition control techniques to underactuated aerial robots.

The contributions discussed in this report satisfy the research goals laid out at the beginning of this project. The dynamic contact transition problem is explored from the lens of a specific use case on aerial robots. The exploration was limited to simulations, an ideal next step would be to validate the proposed solution through physical experiments.

Acknowledgment

I would like to use this space to acknowledge the contributions of the people who have guided, helped or supported me over the course of my graduate studies. I thank them for their valuable efforts during this time.

Firstly, I would like to thank my supervisors, Prof. Antonio Franchi, Dr. Alessandro Saccon and Dr. Chiara Gabellieri. I express my gratitude to Prof. Franchi and Dr. Gabellieri for taking the time to consider my interests and find a suitable graduation project. It was very interesting to interact with Dr. Saccon and learn about impact-aware robotics and the research undertaken at TU/e. I express my gratitude to Dr. Gabellieri for your valuable insights and constant support throughout the project. I would like to thank Prof. Franchi and Dr. Saccon for providing critical feedback that helped me steer the project in all the right directions. I would also like to thank Amr Afifi and Youssef Aboudorra for helping with the software framework and valuable discussions at critical junctures in the project. I also extend my thanks to Jolanda Boelema-Kaufmann for efficiently handling all administrative matters and being such a cheerful presence in the lab.

I am grateful to my fellow students and members of the RAM lab for their support and discussions over lunch or coffee. Your presence made my time at the RAM lab pleasant and enjoyable.

Lastly, I would like to thank my friends and family for supporting me throughout the process.

Contents

Chapter 1

Introduction

1.1 Interaction in Robotics

Interaction in robotics is when a robot makes (or breaks) contact with an object or surface in its environment either intentionally or unintentionally. There are a number of factors that affect interaction based on the state of the system at the instant of contact. The taxonomy of interaction can be roughly divided into four classes [1]:

- **Kinematic Interaction** can be analyzed with kinematics alone. A manipulator that makes contact with the target at zero velocity can serve as an example of Kinematic Interaction.
- **Static Interaction** usually involves additional static forces.
- **Quasi-static Interaction** involves some dynamics which is small enough to provide accurate conclusions despite neglecting their effect in the analysis.
- **Dynamic Interaction** includes interaction tasks that utilize the forces of acceleration.

Modern robots increasingly perform tasks previously seen as only humanly possible [2]. This is especially important in the logistics sector, riddled with repetitive, mundane tasks that can seem too dull to human workers. Although such tasks are perfect for a robotic system, in their current state they are still much less efficient than a human performing the same task. The reason is the ease with which humans handle dynamic tasks like grabbing things from a moving conveyor belt, throwing, packing, etc. Most robots performing such actions intentionally establish contact at zero velocities, which increases the execution time and the additional deceleration and acceleration required expends higher energy.

The complete sequence of events involved in any interaction task can be characterized into three regions: Free motion space; where the robot is still approaching the target, Transition region; where contact is established, and Contact region; where the contact is maintained while performing other tasks [3]. Achieving and maintaining stable contact while performing interactive tasks, formally known as the stable contact transition problem, has recently been a hot research topic. Position/force hybrid control algorithms solve the stable contact transition problem after a finite number of bounces when impact occurs at non-zero velocities[4, 5]. [6] addresses the problem through a two-layer controller. It consists of an impedance control inner layer providing a stable control structure across the three regions and an admittance control outer layer to regulate the output force. Other control approaches include piece-wise control schemes where different control laws are applied to different state space regions [7, 8, 9]. [10] proposes an energy tank-based approach to ensure the stability of the impedance controller during contact transitions.

1.2 Dynamic Contact Transition

Most interaction tasks undertaken by state-of-the-art robots involve some dynamic forces however, in this scenario, the control of the manipulator is designed to tolerate these dynamics rather than actively exploit them. The Dynamic contact transition problem, on the other hand, seeks to achieve the latter. By this definition, the problem is quite complex to analyze but has the potential to increase the repertoire of actions a manipulator may perform. It can expand the range of the manipulator outside their loading capacities and workspaces, thereby simplifying the design requirements for a given task [1].

Several recent works try to overcome the problems encountered in the earlier hybrid controllers and achieve dynamic contact transition. Reference Spreading Hybrid Control (RSHC) tackles hybrid systems with state-triggered jumps through a novel error notion. The error notion helps avoid peaking due to non-coinciding jump times by extending the ante- and post-impact versions of the reference trajectory such that the state of the system is compared to a version of the reference state that has completed the same number of jumps. RSHC can be easily extended to non-periodic trajectories. In [11] RSHC is shown to easily extend to partially inelastic and inelastic impacts. It has also been employed to control robots with flexible joints successfully [12]. [13] also describes a time-invariant version of RSHC where ante- and post-impact trajectories are state-defined. [14] temporarily projects the tracking controller to an impact invariant subspace.

Although highly desirable there are several under-researched areas that prevent implementing dynamic contact transition in robots.

Hardware Infrastructure

Interactions at non-zero velocities cause high impact forces at the robot joints. Rigid robots are not designed to handle these forces and as a result, may break. Modern robots are better equipped to handle interaction through their back-drivable and compliant joints. Compliant joints are better at filtering out impact torques and therefore capable of providing physical protection to the reduction drives. Current hardware designs are progressing towards better handling Dynamic contact transitions but they are far from optimal.

Predictive Capability

Effective control of any robot involves certain predictions. They are further complicated in dynamic contact transition tasks due to rapid changes observed in the state of the system at the moment of contact. The post-impact state prediction is dependent on the ante-impact state, the type of impact, and the coefficient of restitution. Current literature has just started exploring effective methods of calculating these values and providing accurate predictions [15, 16, 17].

Time Mismatch

Another problem faced when controlling robots performing dynamic contact transition is the inevitable time mismatch between the nominal and actual impact events. The mismatch may be due to inconsistencies in the model, errors in state estimation, or inaccuracies in the sensors and actuators. Regardless of the source, this mismatch in timing leads to high errors. The errors eventually converge on their own, however traditional controllers' reaction to the high error results in a phenomenon known as Peaking for single-point-of-contact impacts and more complex behaviour in simultaneous impacts. Recent works like [18, 19, 11, 2, 12, 13] explore controllers that are designed to deal with time mismatch in manipulators and humanoids.

Aim aware collision monitoring

Collision Monitors are usually designed to monitor workspaces and avoid any possible collisions. For designing impact-aware robotic systems, the controller should be capable of distinguishing between intended and unintended impacts and determining a suitable reaction for the robot.

Impact posture planners

Impact posture planners determine the configuration or posture of the robot at the time of impact. There is some inherent danger involved in an impact whether intentional or otherwise. Given specific user-defined inputs like the speed or angular velocity at the time of impact, a posture planner can calculate certain safe configurations that minimize the effect of impact while achieving the goal of the task.

impact-aware robot learning

Autonomous manipulation in tasks involving intended impacts would require that the robot involved learns how to safely deal with impacts in the face of the numerous challenges of the dynamic contact transition problem. In [20], a learning-by-demonstration framework is developed for controlling a robot establishing contact at non-zero velocity. This framework is especially useful for tasks involving simultaneous impacts as it involves multiple impacts at different locations with small time delays making it hard to predict the state during the time interval of impact.

Software Infrastructure

Finally, a well-developed research field would also cater to complex scenarios involving multiple arms or multiple objects to impact. Such a scenario would require dedicated software infrastructure that can effectively detect, plan and control these impacts while performing all the necessary tasks. Oftentimes, the scenarios will have to be tested using realistic simulations and provide data relevant to the task. This additionally may require a dedicated physics engine.

Traditionally, robotic manipulators have been the main focus from an application perspective for the contact transition problem. The tasks vary from impacting a rigid surface [15, 16] or a soft surface [6], sliding on a surface [3, 4] or picking up objects with an end-effector [17], or using a dual-arm manipulator [12]. In recent years, the contact transition problem has also been extended towards humanoids. [18] chooses the hopping motion of the chosen robotic leg while the humanoid in [19] repeatedly makes and breaks contact with a wall at a given distance while balancing on one foot. In [14] the contact transition problem is applied to running tasks on a bipedal robot.

1.3 Research Goal

The brief exploration undertaken in this Chapter has shown that dynamic contact transition in itself is an under-researched field. Prior research undertaken on fixed-base robotic manipulators and humanoids presents various avenues of exploration into dynamic contact transition using aerial robots. However, a systematic approach is required. The main goal of this thesis can then be formulated as the following three research questions:

- ***Is an impact-aware approach interestingly motivated by an application in aerial robotics?***
- ***What distinguishes the aerial robots in this scenario from other robotic systems, i.e., which are the area-specific challenges that arise when applying impact-aware control methods to aerial robots?***
- ***What constitutes a suitable solution to control an aerial robot subject to impacts?***

1.4 Report Outline

The thesis is divided in the following manner. Chapter 2 introduces the background theory like the geometric UAV controller, the wrench observer, and the setup of the simulation environment. Chapter 3 investigates the motivation behind extending the impact-aware approach to aerial robots. It further defines the problems encountered by using the geometric controller for dynamic contact transition tasks. The control strategy developed to deal with the observed post-impact behaviour is detailed in Chapter 4. Chapter 5 discusses various tests that validate the developed control strategy. Finally, Chapter 6 presents the conclusions and recommendations for future research directions.

Chapter 2

Preliminaries

This chapter discusses certain preliminary material required to understand the rest of the work. Section 2.1 discusses the geometric controller used by the UAV. Section 2.2 discusses the wrench observer used later in the solution and Section 2.3 discusses the general software architecture behind the simulations.

2.1 Tracking Control of the Quadrotor in SE(3)

The tracking controller used here is inspired by the controller presented in [21]. Linear control systems for quadrotor UAVs are usually based on Euler angles [22, 23]. However, singularities occur when the UAV is designed to follow complex trajectories. Geometric control avoids this scenario by treating the configuration space of the UAV as a manifold. Therefore the controller can achieve global asymptotic stability. The controller described in this section is a geometric controller that expresses the dynamics of the UAV in the Special Euclidean Group, SE(3) configuration manifold. The UAV CoM is made to follow a given desired trajectory and heading direction. The design further ensures that the three translational and rotational DoF are stabilized by controlling the four propeller speed inputs.

The general taxonomy of a MAV is given in Section A. The inertial reference frame or the world frame, F_W is represented as $\vec{x}_W, \vec{y}_W, \vec{z}_W$ and the body fixed frame, F_B as $\vec{x}_B, \vec{y}_B, \vec{z}_B$. Table 2.1 defines various variables that the controller uses.

The thrust generated by each propeller is assumed to be directly controlled and in a direction normal to the quadrotor plane. Thus the rotor and propeller dynamics are neglected. Oftentimes, e_i is used to select a particular axis. Here, $e_1 = [1; 0; 0], e_2 = [0; 1; 0], e_3 = [0; 0; 1] \in \mathbb{R}^3$. The force allocation matrix, derived in Section A, shows that the total thrust only acts along $\vec{z}_B$. Therefore, in the inertial frame, the total thrust is given by $f_{ctrl}Re_3 \in \mathbb{R}^3$. Given these assumptions, the total thrust and moment become:

$$\begin{bmatrix} f_{ctrl} \\ \tau_{ctrl} \end{bmatrix} = \begin{bmatrix} c_f & c_f & c_f & c_f \\ 0 & c_f d & 0 & -c_f d \\ -c_f d & 0 & c_f d & 0 \\ c_\tau & -c_\tau & c_\tau & -c_\tau \end{bmatrix} u_\lambda \tag{2.1}$$

The determinant of the partial allocation matrix is equal to $-8c_f c_\tau d^2$. Therefore, given that $c_f \neq 0, c_\tau \neq 0, d \neq 0$, it has a rank of 4 and is invertible. The input, u_λ, is obtained by inverting the partial allocation matrix for a set of desired f_{ctrl} and τ_{ctrl}. Therefore f_{ctrl} and τ_{ctrl} are good candidates as control inputs for this controller.

The dynamics of the UAV is given by:

$m \in \mathbb{R}$	The total mass
$J \in \mathbb{R}^{3\times3}$	Inertia matrix in body-fixed frame
$R \in SO(3)$	Rotation Matrix from body-fixed frame to world frame
$\Omega \in \mathbb{R}^3$	Angular velocity in body-fixed frame
$p \in \mathbb{R}^3$	Position of UAV CoM in world frame
$v \in \mathbb{R}^3$	Velocity of UAV CoM in world frame
$d \in \mathbb{R}$	Length of UAV CoM arms
$f_i \in \mathbb{R}$	Thrust generated by the ith propeller about z_B axis
$\tau_i \in \mathbb{R}$	Torque generated by the ith propeller about $-z_B$ axis
$f_{ctrl} \in \mathbb{R}$	The total thrust
$\tau_{ctrl} \in \mathbb{R}^3$	The total moment expressed in body-fixed frame

Table 2.1: Variable Definitions

$$\dot{p} = v \tag{2.2}$$

$$m\dot{v} = f_{ctrl}Re_3 - mge_3 \tag{2.3}$$

$$\dot{R} = R\tilde{\Omega} \tag{2.4}$$

$$\tau_{ctrl} = J\dot{\Omega} + \Omega \times J\Omega \tag{2.5}$$

$(\tilde{.})$ maps $\mathbb{R} \to so(3)$. so(3) is the Lie Algebra of $SO(3)$ and represents the skew-symmetric version of $(.)$. Additionally, $\tilde{x}y = x \times y \;\forall\; x, y \in \mathbb{R}^3$

The controller is designed such that UAV CoM follows a desired trajectory, $p_{des}(t)$ and maintains a desired direction for the first body fixed axis, $\vec{x}_{B_{des}}$. Figure 2.1 illustrates the overall structure of the controller. The outer loop is the position controller. It takes the desired trajectory, $p_{des}(t)$, as the input and calculates the total thrust, f_{ctrl}, and the desired direction of this thrust. Since the total thrust in the inertial frame is given as $f_{ctrl}Re_3$, the direction of the total thrust Re_3 is along the third body fixed axis, $\vec{z}_{B_{des}}$. The desired first and third body fixed axes, $\vec{x}_{B_{des}}, \vec{z}_{B_{des}}$ are then passed on to the inner loop which first calculates the desired attitude, $R_{des} \in SO(3)$ which is then passed on to the attitude tracker. Given that $\vec{x}_{B_{des}}$ and $\vec{z}_{B_{des}}$ are not parallel to each other, $\vec{x}_{B_{des}}$ is projected on a plane normal to $\vec{z}_{B_{des}}$. The remaining DoF in the desired attitude is calculated as the unit vector that is normal to both $\vec{z}_{B_{des}}$ and Proj($\vec{x}_{B_{des}}$). The attitude tracker computes the desired moment, τ_{ctrl}. f_{ctrl} and τ_{ctrl} are then passed on to the Motor Control block to calculate the desired inputs u_λ. As $t \to \infty$, $p(t) \to p_{des}(t)$ and Proj($\vec{x}_B$) $\to$ Proj($\vec{x}_{B_{des}}$). Proj(.) is the normalized projection on a plane orthogonal to $\vec{z}_{B_{des}}$.

2.1.1 Tracking Errors

The tracking controller described above requires tracking errors defined for all tracked states, namely: p, v, R, Ω. The tracking errors in the position and velocity of the CoM are given as follows:

$$e_p = p - p_{des} \tag{2.6}$$

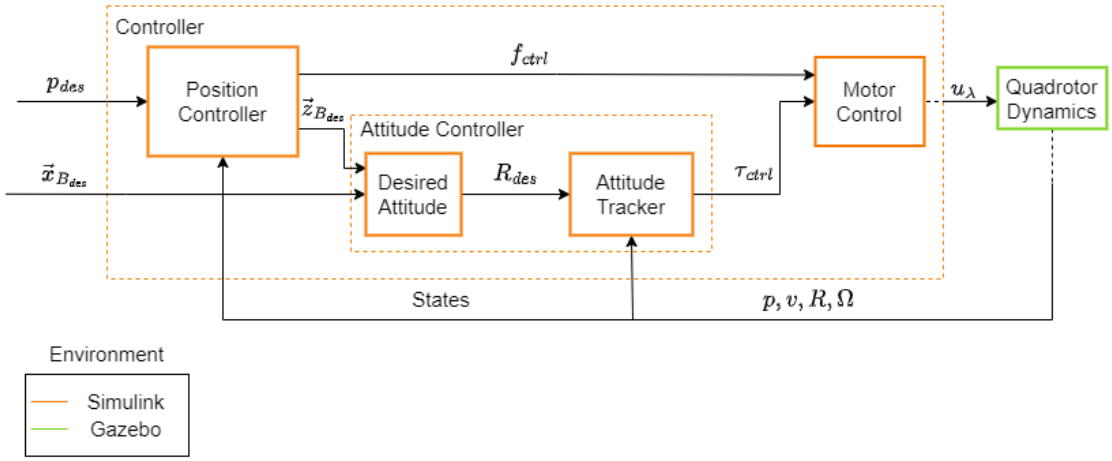


Figure 2.1: Controller Structure

$$e_v = v - v_{des} \tag{2.7}$$

Since the attitude of the UAV is defined in $SO(3)$, the attitude and angular velocity tracking errors evolve around the tangent bundle of $SO(3)$. The error function on $SO(3)$ is given as:

$$\psi(R, R_{des}) = \frac{1}{2} tr[I - R_{des}^T R] \tag{2.8}$$

It can be seen that Equation 2.8 is positive definite when the rotation angle between R and R_{des} is bounded between $[0, 180)$ [21, 24]. Assuming that the variation of the rotation matrix is expressed as $\delta R = R\hat{\eta}$ for $\eta \in \mathbb{R}^3$, the derivative of the error function then becomes:

$$D_R\psi(R, R_{des}) \cdot R\hat{\eta} = -\frac{1}{2} tr[R_{des}^T R\hat{\eta}] \tag{2.9}$$

Using the facts that $-\frac{1}{2}tr[\hat{x}\hat{y}] = x^T y$ for any $x, y \in \mathbb{R}^3$ and vee map $.^\vee : \mathfrak{so}(3) \to \mathbb{R}^3$ is the inverse of the hat map, the attitude error, e_R can be defined as:

$$e_R = \frac{1}{2}(R_{des}^T R - R^T R_{des})^\vee \tag{2.10}$$

The tangent vectors $\dot{R} \in T_R SO(3)$ and $\dot{R}_d \in T_{R_{des}} SO(3)$ lie respectively in the tangent spaces of $R \in SO(3)$ and $R_{des} \in SO(3)$. In order to perform a comparison, one has to transform $\dot{R}_{des}$ into the tangent space, $T_R SO(3)$:

$$\dot{R} - \dot{R}_{des}(R_{des}^T R) = R\hat{\Omega} - R_{des}\hat{\Omega}_{des} R_{des}^T R = R(\Omega - R^T R_{des}\Omega_{des})^\wedge \tag{2.11}$$

The tracking error of the angular velocity is then defined as:

$$e_\Omega = \Omega - R^T R_{des}\Omega_{des} \tag{2.12}$$

2.1.2 Controller Design

Given that the tracking commands, namely desired CoM position, $p_{des}(t)$ and heading, $\vec{x}_{B_{des}}(t)$ are smooth and some gain constants, K_p, K_v, K_R, K_Ω, the direction of the desired thrust, $\vec{z}_{B_{des}}$ can be defined as:

$$\vec{z}_{B_{des}} = -\frac{-K_p e_p - K_v e_v - mge_3 + m\ddot{p}_{des}}{\| - K_p e_p - K_v e_v - mge_3 + m\ddot{p}_{des}\|} \tag{2.13}$$

The underlying assumptions of this definition are:

- $\| - K_p e_p - K_v e_v - mge_3 + m\ddot{p}_{des}\| \neq 0$
- $\vec{x}_{B_{des}}$ is not parallel to $\vec{z}_{B_{des}}$

The desired attitude, R_{des} then becomes, $R_{des} = [\vec{y}_{B_{des}} \times \vec{z}_{B_{des}}, \vec{y}_{B_{des}}, \vec{z}_{B_{des}}] \in SO(3)$. Here, $\vec{y}_{B_{des}} = \vec{z}_{B_{des}} \times \vec{x}_{B_{des}} / \|\vec{z}_{B_{des}} \times \vec{x}_{B_{des}}\|$.

The control inputs can consequently be chosen as:

$$f_{ctrl} = -(-K_p e_p - K_v e_v - mge_3 + m\ddot{p}_{des}) \cdot Re_3 \tag{2.14}$$

$$\tau_{ctrl} = -K_R e_R - K_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^T R_{des} \Omega_{des} - R^T R_{des} \dot{\Omega}_{des}) \tag{2.15}$$

The control moment, τ_{ctrl} is designed such that it performs tracking control on $SO(3)$. For the attitude dynamics defined in Equations 2.4 and 2.5, the controller settles at a zero equilibrium of the tracking errors. Similarly, the control force, f_{ctrl} corresponds to a tracking controller of the dynamic equations; 2.3. Due to the underactuated nature of the system, the tracking errors of the translational dynamics only converge to zero when the attitude tracking error is zero. An increase in attitude tracking errors, on the other hand, implies a large deviation from the desired thrust direction and can lead to instability in the system dynamics. Such a scenario is avoided by design in Equation 2.14. f_{ctrl} is defined as the dot product between the desired thrust direction and the current thrust direction. Thus, deviations between these two vectors are translated into a drop in the total exerted thrust magnitude.

2.1.3 Saturation

The practical implementation of the above controller requires certain extra measures. These measures vary from failsafe measures to considerations as a result of certain characteristics of the vehicle in question. The following section provides an overview of the various saturation measures in place in the implementation of the geometric tracking controller used in this case.

Position Error Saturation

The underactuated nature of the system implies that the translational- and attitude-tracking errors are interdependent. Therefore, a large position error would result in a higher deviation between the desired and current thrust directions which would further increase the tracking errors. As a failsafe, the position error saturation limits the error fed back into the system to a certain empirically determined limit. This measure gives the system time to correct itself without losing stability at the cost of a slower system in the face of large position tracking errors.

Force Saturation

Any given quadrotor is limited by the range of the thrust its motor is capable of producing. Therefore it is important to design the controller to operate within this safe range. This is done by limiting the desired acceleration in the following manner:

$$-\frac{4f_{max}}{m} cos(\theta) \leq \dot{v}_{x,y} \leq \frac{4f_{max}}{m} cos(\theta) \tag{2.16}$$

$$\frac{4f_{min}}{m} - g \leq \dot{v}_z \leq e_3^T (\frac{4f_{max} * R}{m} - g)e_3 \tag{2.17}$$

Here, f_{min} and f_{max} are the minimum and maximum thrusts, respectively, that an individual motor can produce, θ is the maximum angle subtended between mg (g being the gravitational constant) and the maximum total thrust vector produced by the motors. θ is dependent on the mass of the quadrotor used.

Torque Saturation

Similarly, the desired torque needs to be saturated such that it falls within the operational constraints of the motor. However, unlike in the case of the forces, the relationship between the total torque and the motor velocities is not straightforward. The limits here are defined in terms of the forces using the relation:

$$f_{min} \leq c_f(Q \setminus \begin{bmatrix} f_{ctrl}, & \tau_{ctrl} \end{bmatrix}) \geq f_{max} \tag{2.18}$$

Here, Q is the partial allocation defined in Equation 2.1. τ_{ctrl} is limited by the torque generated at the robot's CoM when a thrust of f_{ctrl} is generated by its motors using the classical equation:

$$|\tau| = d|f|sin(\phi) \tag{2.19}$$

Here, d is the length of the quadrotor's arm. ϕ depends on the instantaneous attitude of the rotor and is therefore empirically determined at each instance of the simulation such that it satisfies Equation 2.18.

2.2 Wrench Estimation

The interaction wrench between the tooltip of the aerial robot and the environment can better characterize the physical interaction during an impact. To this end, the wrench estimator proposed in [25, 26] is implemented.

The external wrench or the wrench exerted by the environment on the tooltip, w_E as viewed from the robot CoM, $w_B = \begin{bmatrix} f_B^T, & (R\tau_B^B)^T \end{bmatrix}^T$, is given as:

$$w_B = H_E^T(R) w_E, \; H_E(R) = \begin{bmatrix} I_3 & -[Rp_E^B]_\times \\ O_3 & I_3 \end{bmatrix} \tag{2.20}$$

This implementation further assumes accurate measurements of the robot's position, velocities, both angular and linear, and linear acceleration are available. This is the case for all simulation runs undertaken in this work. An acceleration-based observer is used to estimate the resulting interaction forces, f_B while a momentum-based observer provides the estimated values of the interaction torque, τ_B^B

2.2.1 Estimation of Contact Forces

The external contact force can be obtained directly from the dynamic equation 2.3. However, the acceleration measurement is corrupted by sensor noise. By applying a stable first-order filter, the estimated dynamics is given as [26]:

$$\dot{\hat{f}}_B = L(f_B - \hat{f}_R = -L\hat{f}_B + L(m\ddot{p} + mge_3 - Rf_{ctrl}) \tag{2.21}$$

$L \in \mathbb{R}^{3\times3}$ is the gain matrix of the filter and $\hat{f}_R$ is the estimate of f_R. Given the observer error is defined as $e_f = f_R - \hat{f}_R$, the error dynamics will exponentially converge to the origin for any positive definite matrix L. However, the convergence is contingent on the presence of a constant or slowly varying external force. [25]

2.2.2 Estimation of Contact Torques

The contact torques are estimated by observing the angular momentum $q^R \in \mathbb{R}^3$ in the body-fixed frame of the robot. Rewriting 2.5 in terms of q^R gives:

$$\dot{q}^B = J\dot{\Omega}_B^B = -\Omega_B^B \times J\Omega_B^B + \tau_{ctrl} + \tau_B^B \tag{2.22}$$

The estimate, $\hat{\tau}_B$ can be defined as residual vector [25]:

$$\hat{\tau}_B^B = K_I \left[(q^B(t) - q^B(t_0)) + \int_{t_0}^{t} (\Omega_B^B \times J\Omega_B^B - \tau_{ctrl} - \hat{\tau}_B^B) d\tau \right] \tag{2.23}$$

t and t_0 are the current and initial time instants and K_I is a positive definite gain matrix. By differentiating 2.23, the residual dynamics becomes:

$$\dot{\hat{\tau}}_B^B + K_I\hat{\tau}_B^B = K_I\tau_B^B \tag{2.24}$$

This first-order low-pass dynamic system converges such that $\hat{\tau}_B^B \to \tau_B^B$ when $t \to \infty$ for any positive definite matrix K_I. It is to be noted that the choice of K_I offers a trade-off between the convergence rate and noise-filtering properties of the observer. Higher K_I values lead to faster convergence while lower values filter out the high-frequency components of the noise signal.

2.3 The Simulation Environment

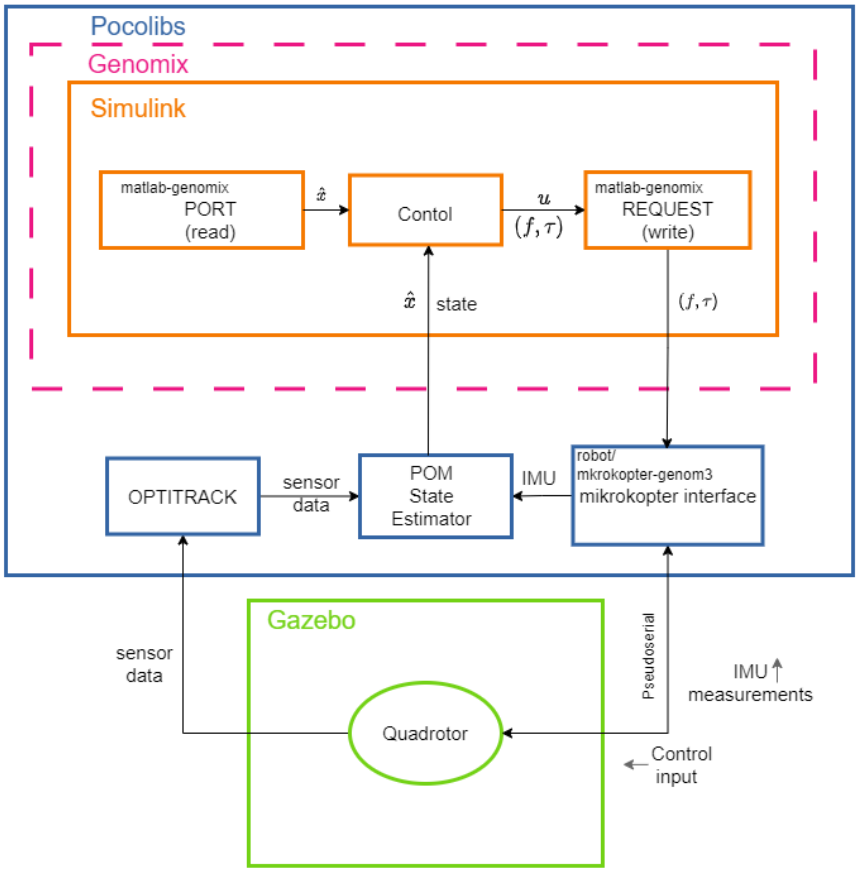


Figure 2.2: Software Architecture; the blocks are coloured according to the environment where they run.

Figure 2.2 provides an overview of the main software architecture used to run all simulations in this work. The geometric controller described in Section 2.1 is set up in Matlab/Simulink. The controller interfaces with several Generator of Modules (GenoM) modules and controls the robot simulation spawned on Gazebo. Generator of Modules (GenoM) generates software components that ensure that different middleware is compatible with one another. The interface facilitating this with Matlab is called matlab-genomix. The ports PORT and REQUEST enable communication between GenoM modules in Matlab/Simulink and those that provide information about the robots and the simulation. The packages used here are:

- **Optitrack** retrieves the pose of the robot. ie. the position and orientation of the UAV from Gazebo.
- **POM** estimates the state of the robot using the information provided by the sensors. The estimation is provided at a rate of 1 kHz
- **Mikrokopter interface** interfaces with on-board flight controller. It accepts desired thrust and torque values as input and commands the robot on Gazebo through pseudoserial communication.
- **Port** and **Request** Simulink blocks respectively read and write to the external blocks. The communication uses the service genomix.

Chapter 3

Motivation for Impact-Aware Approach in Aerial Robotics

This chapter examines the main motivations behind applying the impact-aware approach to aerial robots. Section 3.1 discusses the current state of the art in aerial robotics specifically in aerial manipulation. Section 3.2 explores the scope of extending the dynamic contact transition problem to aerial robots. Lastly, Section 3.3 motivates the need for an impact-aware approach by testing the limits of the geometric controller when performing dynamic contact transition tasks. Thereby, this section further introduces the specific problem addressed in this work.

3.1 Aerial Robots: State of the Art

Aerial robots have grown in popularity owing to a continuous extension of their use cases. Traditionally, they were used solely as a sensing platform incapable of physical interaction with their environment [27, 28, 25, 29, 30]. Major points of interest when attempting to establish physical interaction between robotic devices and their surrounding environment involve autonomous behaviour that ensures safe interaction in unstructured environments without the influence of human skills. Several factors complicate facilitating interaction between aerial robots and their environment. Most popular aerial robots are underactuated, this implies that the control of a given platform's position is coupled with the control of its attitude and vice versa. Additionally, when a manipulator is mounted on an aerial robot handling impacts is inherently harder due to the floating base. The forces and torques experienced by the manipulator affect the UAV's position. The performance of the propulsion system is further affected by the aerodynamic effects experienced close to a given surface during an interaction. Stringent payload requirements constrain the type of manipulator that can be loaded on a given system without compromising on agility [27, 31, 32].

Due to the unique set of problems encountered, Aerial manipulation is considered a separate research theme concerned with systems that include a Unmanned Aerial Vehicle (UAV) and an interaction mechanism. Quadrotors are the most widely used platforms for aerial manipulation due to their simple mechanical design, low cost, agility, and abundance of existing literature on control schemes. Even though there are works like [25, 33, 34] that describe fully actuated systems that offer better control and stable hovering, they consume more energy and would in turn only contribute to a niche percentage of the market.

Review papers like [31, 27] identify four main categories of physical interaction, namely: Unmanned Aerial Vehicles (UAVs) equipped with a fully actuated robotic manipulator, a gripper, a rigid tool attached directly to the body, or tethers. In terms of missions, load transportation is the most common task for aerial manipulators. The load is typically hung below the UAV by a tether or picked up by the gripper/manipulator. A second type of mission involves the manipulators per-

forming tasks requiring force/torque exertion to the environment like inspection, valve turning, and door opening. Other categories of missions include assembly and structural construction (see [31] and references therein). Two kinds of operational scenarios are seen for aerial manipulation tasks. A sequential scenario is the most common approach where the flight and manipulation are controlled independently. The concurrent scenario is harder to optimize but helps reduce operation time and improve hovering and tracking performances by exploiting the coupled dynamics [31].

3.2 Impact-Aware Aerial Robots

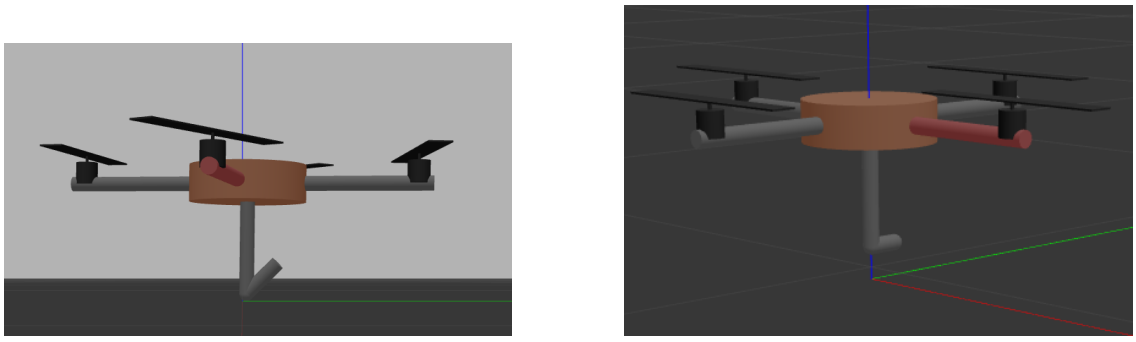
The treatment of the dynamic contact transition problem in aerial robotics can be seen as a special case of aerial manipulation. While any of the above-discussed categories of physical interaction may be studied, the missions, in this case, are limited to tasks that involve some kind of impact that can be performed dynamically. This excludes missions like the one where a given load is transported by a tether. Additionally, by the nature of the task, the problem only considers the concurrent operation scenario. The new field emerging from this classification is termed impact-aware aerial robotics.

Given the advancements in the design of aerial robots, much like the discussion regarding fixed-based manipulators, the hardware of aerial manipulators is in a position to implement dynamic contact transition. Owing to the nearly unlimited workspace, aerial robots capable of Dynamic contact transition have many applications. In the long term, dynamic contact transition can extend robotic manipulation capabilities to any task involving heights like the interception of flying targets. It could further perform tasks such as swooping seen in birds and explore locomotion techniques such as leaping. Tasks like swooping have numerous practical applications especially related to picking up objects from non-solid surfaces or surfaces deemed dangerous to the system from prolonged contact. Another possible application is pushing heavy objects by dynamically exerting force. A first step towards its realization would be stabilizing the UAV while undergoing Dynamic contact transition. Despite the advances in hardware reliability and the extension of use cases for aerial robots, collision exploitation is still an uncharted research territory. [35, 36] propose a system that can capture airborne targets using a vision-based system. However, the real interaction between the target and the manipulator is not dynamic since the target is followed around before being intercepted. [33, 34] use UAVs to exert force on a target. However, these works achieve it through thrust vectoring using tilted rotors and fully actuated systems respectively. [37] designs an actuated appendage attached to a quadrotor that can perform the swooping motion. In this design, the system is decoupled such that the grabbing motion achieved at a lower speed does not affect the flight of the quadrotor. [38] proposes a water sampling UAV which hovers above the water before performing the sampling. Achieving these tasks dynamically could save vital energy/fuel which is a critical feature of most modern aerial robots. Additionally, dynamic contact transition capabilities can significantly improve the efficiency of picking objects or collecting samples.

3.3 Problem Definition

This work focuses on the general use case of swooping. Swooping is usually seen in predatory birds that spot their prey from a certain altitude, fly down, grab the intended target and fly up in a short span of time. By identifying certain key features of the swooping motion, the motion can be translated into a set of tasks for a UAV. The key features in this sequence of actions are: impacts at relatively low altitudes, impact events resulting in additional payload, impacts occurring at a certain velocity and the desired trajectory continues after the impact event. Therefore the chosen desired trajectory requires the vehicle to fly up to an altitude of 0.75 m (relatively low altitude), and achieve a certain desired translational velocity along the y-axis. The vehicle impacts an object

in its path at this desired velocity and continues with the same velocity while carrying the object. All simulation experiments are conducted on a quadrotor with a hook attached below its CoM as shown in Figure 3.1. The tasks are designed such that the impact event entails single-point-of-contact impacts.



(a) (b)

Figure 3.1: Passive manipulator designs. (a) The hook is designed for carrying tasks like in Loop and H scenarios. The sharp kink ensures single-point-of-contact impacts. (b) This variation is designed for the Block scenario. The rounded 90° tip ensures single-point-of-contact impact in this scenario.

Three different scenarios are examined based on the design of the target being impacted. The target is placed in the path of the quadrotor such that it approaches it at the desired velocity (Figure 3.2).

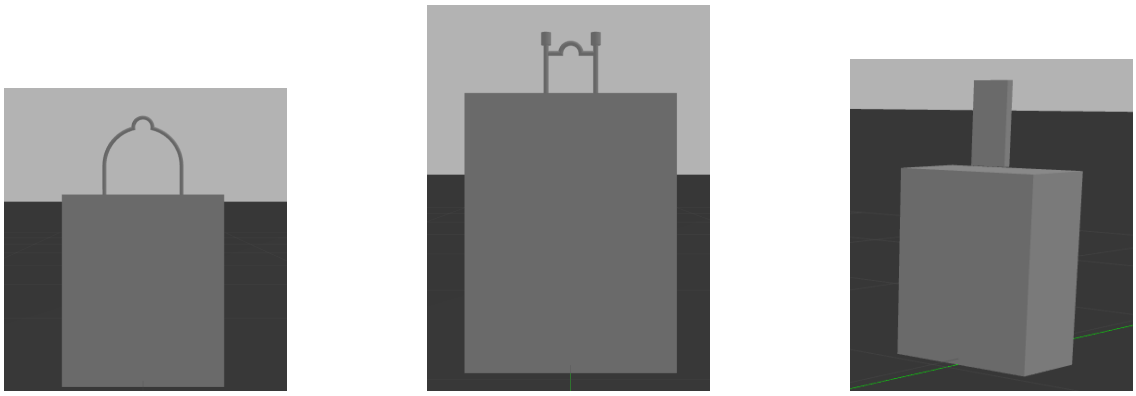
- **Loop** is the scenario that is the closest representation of the use case. The loop is designed like the ear of a basket. The clamp reduces the pendulum-like relative motion between the hook and the loop.
- **H** scenario serves as a control case of the Loop scenario. The H-like design of the object ensures that impact occurs at its CoM and therefore reduces the total torque seen at the quadrotor's CoM following the impact.
- **Block** scenario is another control case. Here, the object is a thin block which topples upon contact. This scenario isolates the impact event (Transition region) from the rest of the task objective (Contact Region).

The iterative scenario selection process is discussed in detail in Appendix B. The system described above fails from a desired velocity of 2.5 m/s when using the geometric controller (Section 2.1).

Irrespective of the velocity at impact, the system always undergoes a perturbation in its attitude. The system either recovers from the perturbation, flies unsteadily for the rest of the run, or crashes. Empirical trends show a higher incidence of crashes for higher velocities at impact. The difference in behaviour for a similar set of inputs can be attributed to slightly different outputs from the Polynomial Trajectory Block in Simulink and small differences in the quadrotor's initial conditions (Appendix D). However, this noise can be seen as adding a degree of uncertainty to the simulation as is the case in reality. Based on the empirical data gathered over multiple simulation runs the following characteristics can be identified as common traits for this problem:

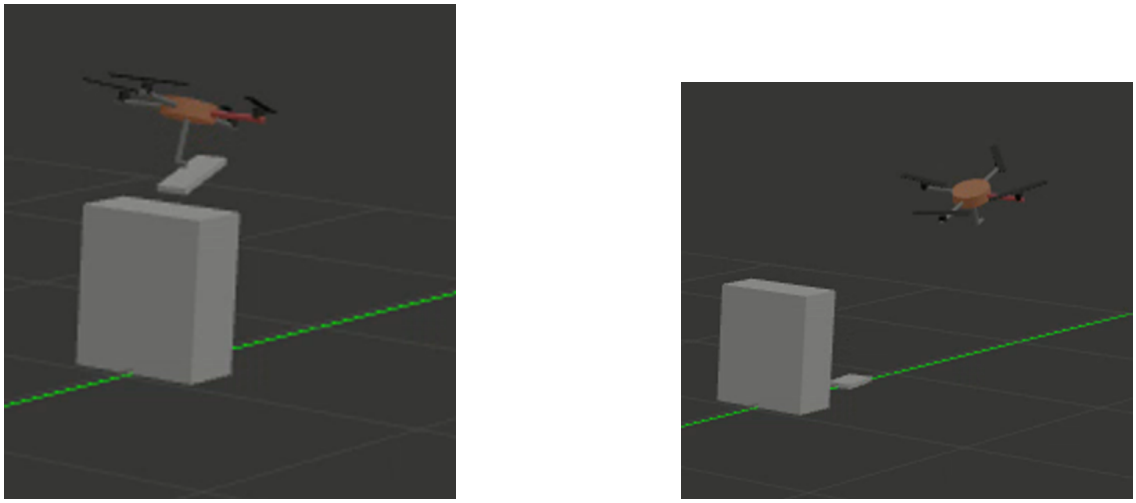
Velocity at Impact Affects Behavior

Irrespective of the nature of the object, the velocity at impact seems to affect the post-impact behaviour in a similar manner. Impacts at lower speeds are more likely to recover from the



(a) Loop scenario (b) H scenario (c) Block scenario

Figure 3.2: Object designs for the three selected scenarios



(a) Forward Tilt (b) Quadrotor tilting back

Figure 3.3: An example of the characteristic forward tilt observed

perturbation while impacts at higher speeds result in crashes.

Forward Tilt

In all cases, the quadrotor tilts forward as seen in Figure 3.3a, leaning in the direction of motion. The magnitude of tilt is proportional to the velocity at which impact occurs. The quadrotor responds to this perturbation by tilting hard in the reverse direction as seen in Figure 3.3b. For the Loop scenario, it was further observed that relative motion between the hook and the object tended to inadvertently aid in the correction of this tilt. This effect is not observed in the H scenario where the design limits this relative motion.

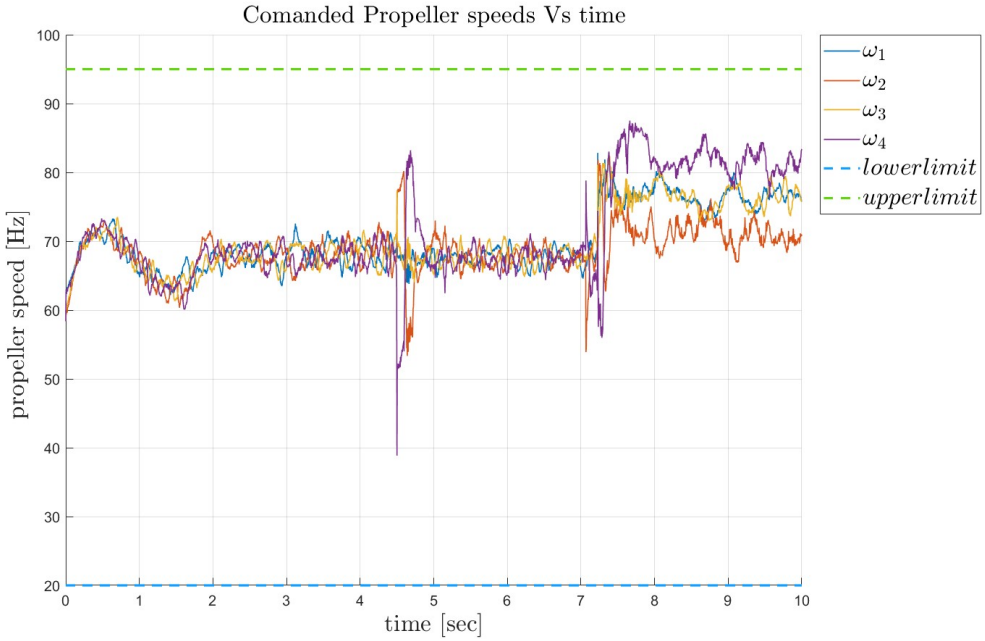
Input Saturation

Crashes occur when the commanded propeller speed, which acts as the controller input, rises/falls toward saturation levels. A noticeable spike is observed in the commanded propeller speeds at the instant of impact. The spike in magnitude is slightly higher for higher speeds at the time of impact. In instances of a crash, this spike leads to further increase (or decrease) in the commanded

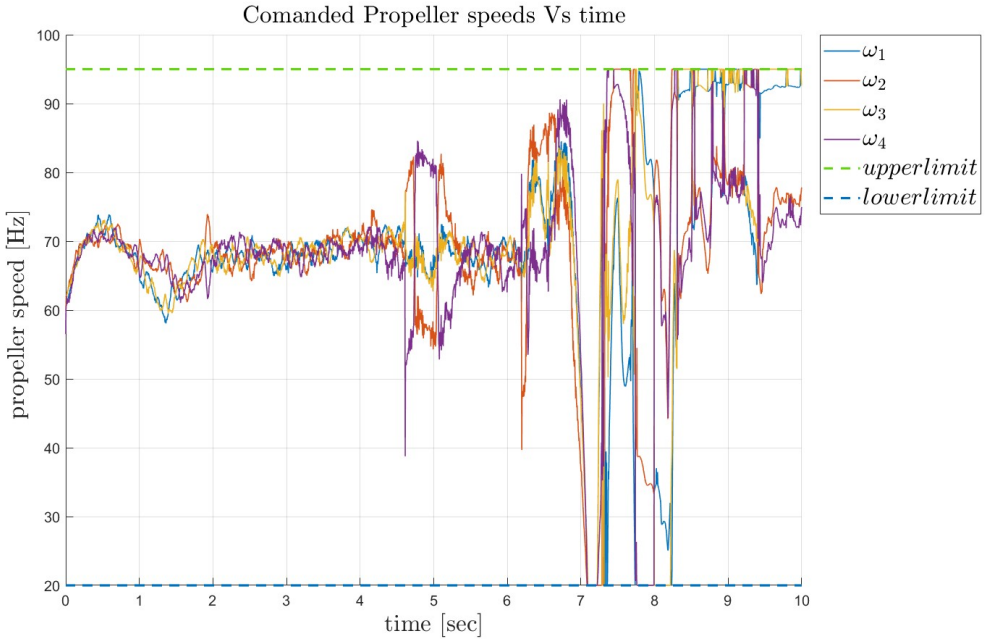
propeller speeds until it reaches the values corresponding to input saturation (95 Hz and 20 Hz respectively) (Figure 3.4).

Of the three traits identified, input saturation is the most interesting. Input saturation occurs when the controller is expected to perform too many tasks at once. Depending on the hardware capabilities, every system has a limit on its task-performing capacity. The term control authority is used from here on to refer to this capacity. During an impact event, a lot of tasks require the controller's attention in addition to the regular tracking tasks. Performing all tasks at once divides up the available control authority causing the system to saturate and eventually fail. In addition, here the impact event causes the quadrotor to tilt forward. The thrust vector is then facing the ground. The low altitude of the selected scenario does not afford a lot of space for recovery in this direction. Therefore the main causes of failure are identified as the low altitude of the impact event and the input saturation of the quadrotor. Detailed analysis of the post-impact event behaviour can be found in Appendix C.

This work seeks to extend the range of desired velocities at which the given quadrotor can perform the selected task. It is to be noted that the range can be easily increased if the input saturation limits are extended/removed. Such a study may hold some theoretical relevance however, in most practical applications UAVs are affected by the input saturation set by the load capacities of the motors used. Therefore an effective solution should incorporate these limits.



(a) Velocity at Impact: 1.5 m/s



(b) Velocity at Impact: 2.5 m/s

Figure 3.4: Example of input saturation as observed through the quadrotor commanded propeller speeds. The dotted lines indicate the saturation limits. (a) The commanded propeller velocities when the impact event is seen as a disturbance by the controller. (b) The commanded propeller velocities when the impact event leads to input saturation.

Chapter 4

Impact-Aware Control Strategy

Input saturation is a common problem encountered when controlling a UAV. Therefore, there are a number of different ways to approach the solution. One option is to redesign the controller to adapt towards the behaviour seen during the impact event. Control techniques like adaptive backstepping control or nested saturation control using an admittance control external loop could better adapt to the post-impact behaviour. Another possible solution is gain scheduling. Here the controller briefly switches to gains tuned for the quadrotor during the impact event. Yet another viable solution to reduce the demand for control authority is to prioritize certain tasks over others until sufficient control authority is available to achieve all the other tasks. Therefore, in this case, the reference objectives of the controller are shaped.

Of the three viable solutions, this work opts for the third approach. The biggest advantage of this approach is that it does not require a complete change of the control algorithm. Although gain scheduling also offers this advantage, tuning gains for the impact event is quite complex from an execution perspective. For one, the impact event and resulting post-impact behaviour occur for a short span of time and cannot be isolated from running the tracking task. This would mean running multiple trials while iteratively calibrating the gains for the impact event. This approach is both time-consuming and inefficient.

The impact-aware control strategy proposed in this work is detailed in the rest of this chapter. Section 4.1 introduces the interaction planning problem that serves as the inspiration for the proposed control scheme. The main objectives of the control scheme are further lined out in this section. The remainder of this chapter discusses the solution proposed for dealing with impacts in the given scenario. The error signals are observed in Section 4.2 in order to determine which factors primarily drive the system to input saturation. These observations are then translated into a priority order for the desired tasks and additional constraints are determined based on the behavior observed. This knowledge is then used to determine an appropriate reflex reaction in Section 4.3. A suitable activation signal is determined to trigger the reflex action in Section 4.3.1. Lastly, a suitable re-entrance strategy is proposed that ensures that the quadrotor safely returns to the nominal task execution mode in Section 4.3.2.

4.1 Interaction Planning Problem

The control scheme introduced in this chapter falls broadly into the category of the Interaction Planning Problem introduced in [39, 40]. The interaction planning problem consists of three main players; the system state of the robot, the state of any human(s) included in the scenario, and the state of the environment, which includes objects involved in the scenario and the state of the abstract task. This information set is collectively referred to as the 'interaction world state', WS, and can be defined as:

$$WS = RS \times HS^n \times OBS^m \times TS \tag{4.1}$$

Here, RS,HS,OBS, and TS respectively denote the robot state, n human states, m objects, and the task state. The problem is greatly simplified in the context of the current scenario. The world state, WS in this case simplifies to:

$$WS = RS \times OBS^1 \times TS \tag{4.2}$$

Each of these states in turn form bigger sets of states that can be further divided. Of these, the robot and task states are of particular relevance in this scenario.

The robot state, RS is defined as

$$RS = S \tag{4.3}$$

This set provides information on the internal state of the robot, S. The internal states set is further subdivided into $S = S_{ac} \times S_b \times S_p$. S_{ac} is the set of actions taken by the robots while S_b is the set of robot behaviour characterized by controller choices and reflex reactions. Lastly, S_p represents the instantaneous physical state of the robot at any given moment.

The task states, TS can be subdivided as:

$$TS = A \times TC \tag{4.4}$$

This set consists of the set of all possible actions, A and the task criticality, TC.

With the above definition, the interaction planning problem is then about selecting the appropriate robot action and behaviour at any given instance in time based on the information enclosed in the world set, WS, and the knowledge base, KB (object properties, laws of physics, etc). This can be expressed in the form of a select action, sa mapping:

$$sa : WS \times KB \rightarrow S_{ac} \times S_b \tag{4.5}$$

Two main actions can be identified for the A in this scenario, namely, trajectory tracking and gravity compensation. The interaction event is the impact between the robot and the object (OBS) in the robot's path. In that instance, the robot should rely on reflex reactions S_b rather than follow the set of tasks, S_{ac} at hand. Therefore, the task criticality, TC in this scenario unlike in [39] should rely on the effect of task failure on the robot's safety (i.e., avoiding crashes).

Just like in the case of humans, robot reflex reactions are self-preservatory actions taken in light of unexpected disturbances that have the potential to disrupt the current state and result in a crash. A reflex reaction overrides the normal set of tasks and their corresponding set of robot actions based on certain environmental or internal factors that stand out from those observed during general task execution. In most cases, a robot reflex is the consequence of an activation signal. The signal indicates the presence of a stimuli or a fault. Once executed the question of re-entry into task execution mode comes into play.

Two lines of queries arise from the above discussion, namely, the suitable reflex reaction to employ in the given scenario and the relevant strategy of re-entry. [26, 41] implement reflex reactions for aerial robots in the context of flying in an unknown possibly cluttered environment. The primary objective of this work was obstacle avoidance and mapping a cluttered environment based on external wrench estimation. Evaluating the reflex reaction schemes discussed in [26, 41] provided some insights into devising a reflex reaction scheme for the particular problem introduced here. The main objectives of the control scheme can therefore be summarized as follows:

- The reflex reaction should dissipate some of the energy gained by the quadrotor as a result of the impact till it recovers from the perturbation.
- Due to the low altitude of the quadrotor at impact, it cannot afford to expend energy in the downward direction as such an endeavour would inevitably result in a crash.

4.2 Error Visualization

Input saturation occurs because the demands on the controller exceed the capability of the system. In order to ensure the safe recovery of the quadrotor after the impact subevent, the controller demand has to be managed efficiently. In other words, a specific priority order has to be realized and the controller should focus on tasks of higher priority till it starts operating within its limits again. Given the tracking nature of the controller, it is fair to assume that high tracking errors drive it toward saturation. Therefore, as a first step, the tracking error signals soon after the impact subevent are observed.

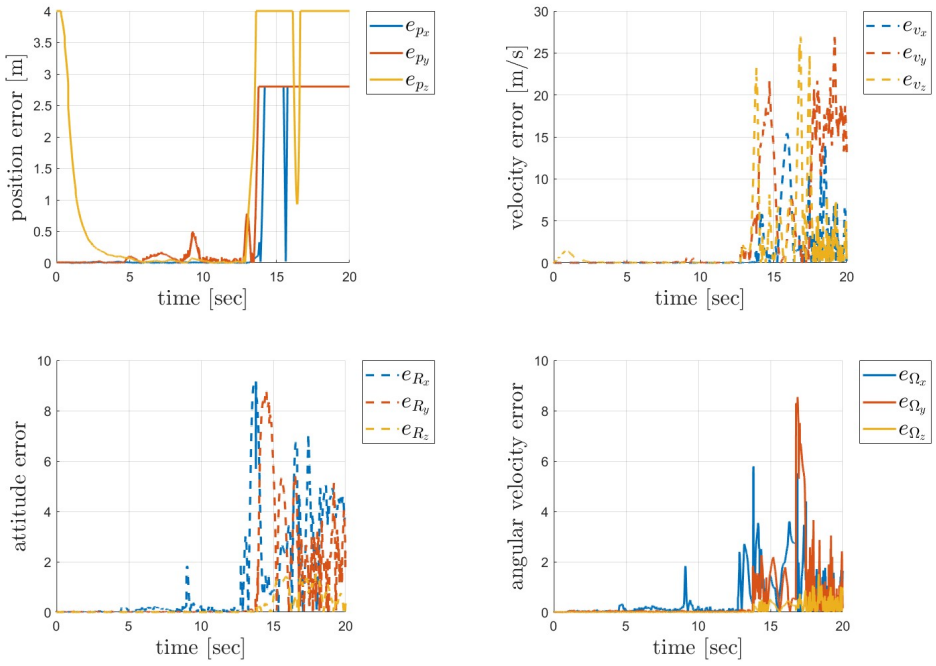


Figure 4.1: Scaled tracking error signals

Figures 4.1 show the scaled tracking errors (scaled on controller tracking gains) from the Loop scenario numerical simulation executed at a desired velocity of 2.5 m/s. The sudden spike in error values around 13 s indicates the occurrence of the impact event. The main contributors driving the controller since that point in the translational tracking errors are the velocity tracking errors along the y and z axis respectively. It is to be noted that this analysis is germane to the simulation scenario design and desired quadrotor trajectory in this case. Since the desired trajectory is two-dimensional along y and z, it is understandable that these two tracking errors are affected the most. There is also a slow rise in the velocity tracking error along x, however, at this point, the quadrotor has already crashed and therefore the data is not quite as relevant. Another interesting point is the saturation in position tracking errors. However, as pointed out in Section 2.1 the position tracking errors are saturated to ensure stability. The difference in the saturated values of

the x, and y position tracking errors to that along the z-axis is due to the different gain values.

In the orientation errors, the main contributors are the attitude tracking error about the x-axis and its corresponding angular velocity tracking error. This corroborates with the quadrotor's post-impact behaviour and the fact that it experiences an impact at a point lower than its Center of Mass. Similar to the translational error case, there is an increase in the other tracking errors after a period of time, however, this rise occurs after the quadrotor crashes and hence loses relevance.

From the above discussion, it is clear that the controller is driven to saturation by the tracking translational errors along the y and z-axes and attitude errors about the x-axis. In the next section, this data is analyzed further to determine a suitable reflex reaction.

4.3 Reflex Reaction

From the error signals, it is clear that the solution involves switching off (or giving up) control in one or more of the degrees of freedom (DoF) identified. The naive approach here would be to switch off translation tracking along the y and z-axes and attitude tracking about the x-axis. But a closer look at the post-impact behaviour would point out the flaws of this approach. When all the above tracking errors are switched off, the controller is now free to move along any of the available degrees of freedom. With no prior knowledge or predictive capabilities, the quadrotor is now free to fly downwards straight towards the ground in the absence of all tracking references. This is clearly undesirable.

Looking at each of the degrees of freedom (DoF) in turn will give an idea about the role they play in post-impact behaviour. Switching off translational tracking along the y-axis will offer the controller a lot of freedom since the desired trajectory is primarily designed along the y-axis. Freeing up translational tracking control along the z-axis, on the other hand, has a counter effect. Irrespective of the simulation conditions, it is essential that the quadrotor at least performs gravity compensation. However, in this case, owing to the close proximity of the desired trajectory to the ground, it is also essential it does not lose altitude. An alternative could be to direct the quadrotor to fly higher, however, this would require using up more control authority. The question of freeing up attitude control along the x-axis is slightly more complicated. In a fully actuated system freeing up control along this DoF would be a viable option. However, in an underactuated system such as this, the attitude tracking error is interdependent with the translational tracking error. Additionally, it was observed that the post-impact quadrotor is tilted about the x-axis such that the thrust direction is facing downwards. If the controller fails to correct this forward tilt, the eventuality is a crash.

Therefore, the best reflex reaction for the given scenario is to switch off tracking control along the y-axis while the z-axis translation and x-axis attitude tracking are closely controlled. Such a strategy has the potential to free up enough control authority so as to keep the controller within its operating limits and effectively recover from the perturbation caused by the impact.

4.3.1 Activation Signal

The next step is to identify a suitable activation signal for this reflex reaction. The activation signal should meet the following criteria:

- The signal either undergoes a change or is inherently related to the impact subevent
- It should not exhibit similar features anywhere else for the duration of a simulation run
- It should be easily measurable
- The measurement of the signal should not cause any additional delay in the system

The external wrench estimate offers a robust indication of an impact event. Additionally, [26] suggests that the frequency characteristics of the signal help differentiate between the aerodynamic and collision-related forces. Therefore, it also offers a robust indication even in environments influenced by wind. As discussed earlier, the signal should meet a set of criteria.

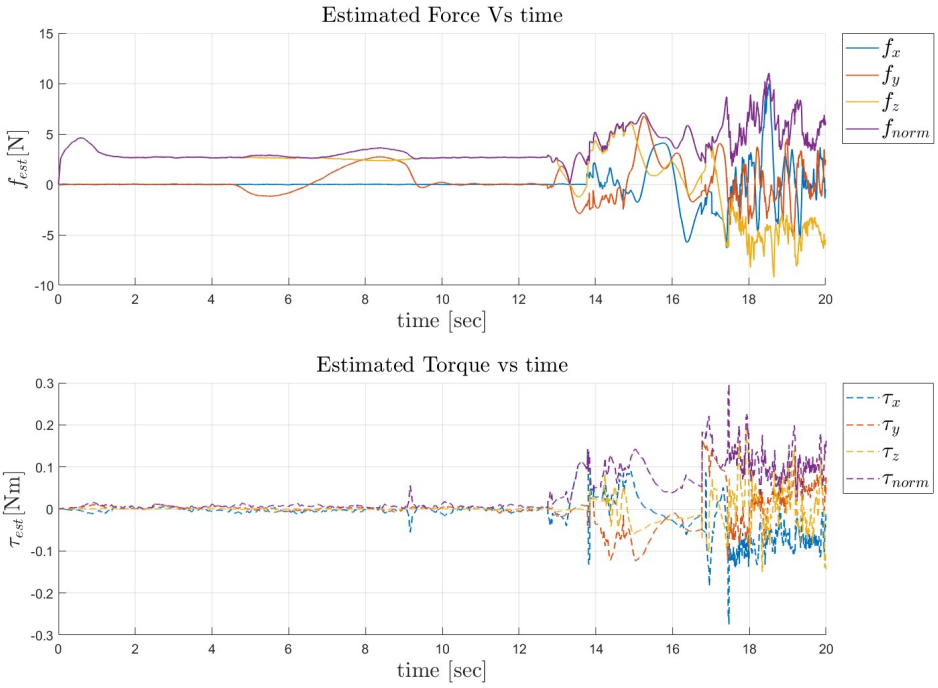


Figure 4.2: Wrench estimate signals for simulations using the geometric controller.

Figure 4.2 shows the raw wrench estimate signals. Although the impact event is discernable from the sudden jump in the signal values, there is no clear threshold. It is to be noted that the external wrench is estimated at the CoM of the quadrotor, however, due to the addition of the hook the position of the CoM of the whole vehicle shifts downwards. Hence, changes in the trajectory (like when it shifts towards a constant velocity trajectory) are also recorded as an external wrench. From the graph, it is evident that the impact event is faster than the other trajectory changes. Therefore, the rate of the wrench estimate signal should better highlight this event.

The rate signal shown in Figure 4.3 uses a Simulink memory block and obtains a signal given as $\tau_{est}(k) - \tau_{est}(k-1)$ for a discrete-time instance, k. It offers consistent thresholds across all relevant simulation scenarios and is therefore chosen as the impact detection signal. The impact detection signal also functions as an ideal activation signal for the Reflex Reaction Mode.

$$\text{Reflex Reaction Mode (RRM)} = \begin{cases} 1, & \text{if } f_{rate} \geq 0.22 \wedge \tau_{rate} \geq 0.01 \\ 0, & otherwise \end{cases} \tag{4.6}$$

4.3.2 Re-entrance Strategy

Two aspects form the crux of the strategy to re-enter the tracking task execution mode. The (de)activation signal and the considerations about the reference trajectory. The factors that determine that the quadrotor has recovered from its perturbation due to the impact provide

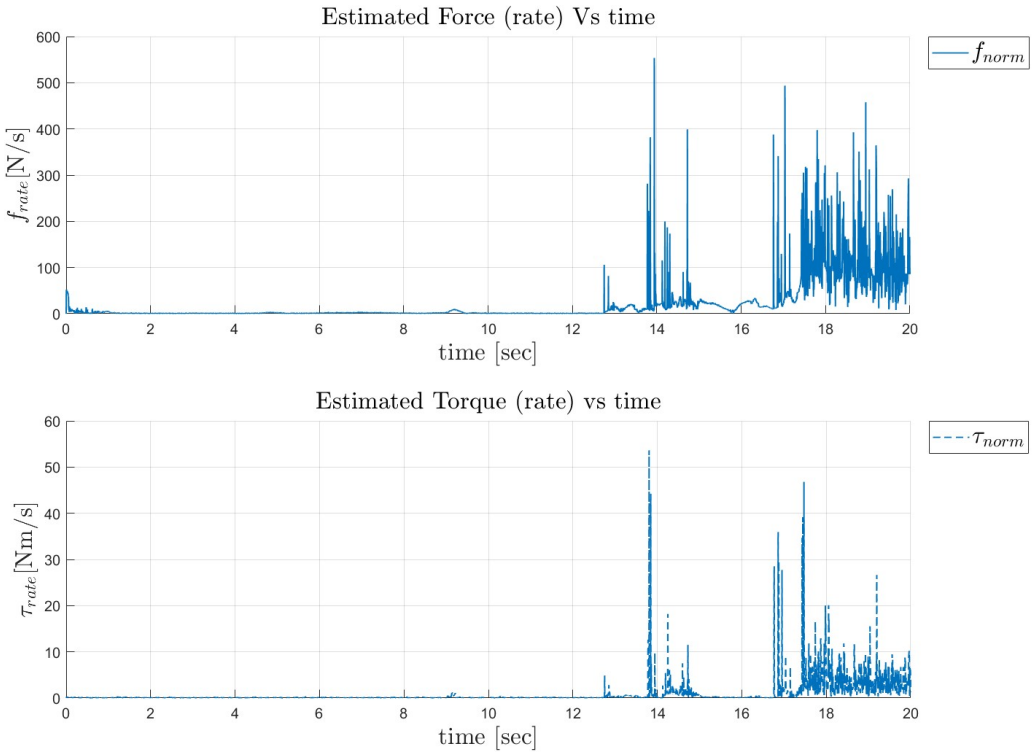


Figure 4.3: Rate of Wrench estimate signals for simulations using the geometric controller. The first peaks appear as a result of the impact event. The second set of peaks occurs after the crash.

insights into the (de)activation signal to use for the re-entrance strategy. The (de)activation signal has a similar set of requirements as the activation signal used to enable the reflex reaction. In addition, this signal should only be activated once the quadrotor is already in the reflex reaction mode. Out of a number of viable signals, one that remained consistent and presented a similar threshold for all relevant simulation scenarios was the difference signal for the external torque estimate.

$$\text{Re-Entrance Mode (REM)} = \begin{cases} 1, & \text{if RRM} = 1 \wedge \tau_{rate} \leq 2 \times 10^{-3} \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

Here, $\tau_{rate} = \tau_{est}(k) - \tau_{est}(k-1)$ for a discrete-time instance, k

The geometric controller assumes the desired reference trajectory is smooth and devoid of any jumps. For the entire duration of the reflex reaction mode, however small, the quadrotor is free to change its state along the y-axis, and therefore its states along the y-axis at the instant of re-entrance will differ from the reference states. Following the original reference trajectory implies a jump and can propel the system towards instability. There are a number of ways to avoid this. The strategy employed here ensures that at the instance of re-entrance, the reference trajectory jumps toward the current state of the quadrotor (along the y-axis) and resumes the constant velocity trajectory albeit at different initial conditions.

This chapter introduced the solution proposed for the problem defined in Chapter 3. The resultant control scheme is inspired by the interaction planning problem and defines a reflex reaction mode that the system enters when an impact occurs. The next chapter further evaluates

the merit of the hypotheses that form the basis of the proposed control scheme and validates the proposed scheme through numerical simulations.

Chapter 5

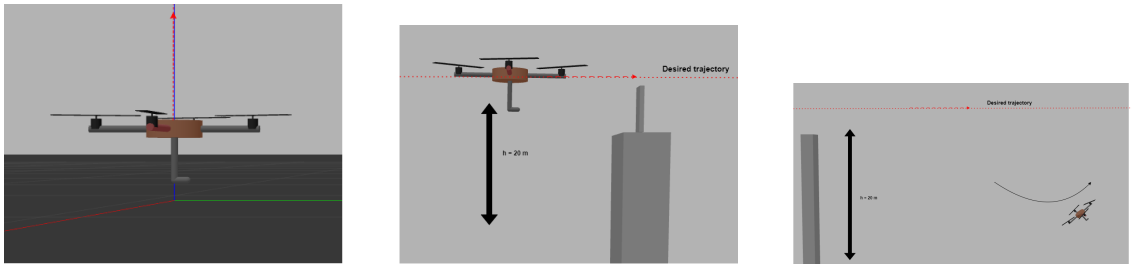
Validation

This chapter details the various validation experiments conducted on the proposed control scheme. Section 5.1 test the hypotheses about the primary causes of the post-impact behaviour observed. Section 5.2 validates the solution through numerical simulation of three selected scenarios and performs a comparison between their performance before and after implementing the solution. Section 5.3 identifies a tracking-error-defined boundary between a disturbance and an impact event. Lastly Section 5.4 explores the impact-invariant subspace in the context of aerial robots.

5.1 Validating the Low-Altitude Hypothesis

While observing the post-impact behaviour it was hypothesized that the height at which the impact event occurs plays a significant role in the behavior. However, it wasn't made clear what role it plays and how significant its role is. This section attempts to provide proof of this hypothesis and better understand the significance of height in post-impact behaviour. This is done empirically by means of a simulation scenario designed for this purpose.

To understand the significance of the low altitude of the impact event, it is important to understand what happens when the impact event occurs at a higher altitude. Figure 5.1 shows three snapshots from the high-altitude simulation. Most elements of the simulation scenario are kept constant. However, the impact event is set at an altitude of 20 m. Therefore, now the object is placed at 20 m. No additional modifications are brought about on the object, instead, the supporting structure providing the desired elevation is tweaked to attain the desired height. The horizontal distance between the initial position of the robot and the object is kept the same. No



(a) (b) (c)

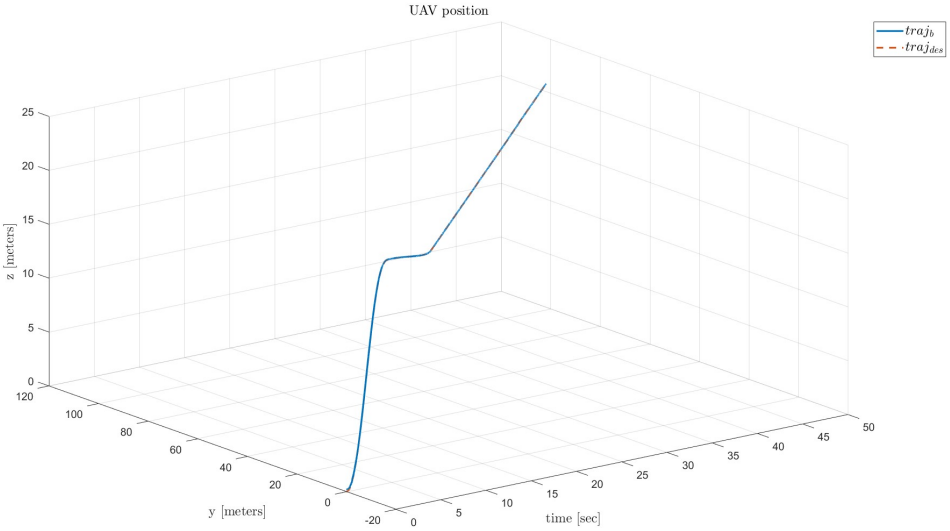
Figure 5.1: Snapshots of a representative high altitude experiment. (a) initial configuration. (b) quadrotor follows the reference trajectory before impact. (c) sample behaviour immediately after impact. The dotted red line represents the desired trajectory and the black solid line in (c) is a representation of the actual path followed by the quadrotor.

additional changes are made to the quadrotor and the spawning process. Further, the general nature of the trajectory remains the same. Once initiated, the quadrotor flies along the z-axis till it achieves the desired altitude (20 m here), then it accelerates along the y-axis till it achieves the desired velocity at impact and continues along the y-axis while maintaining the desired velocity at impact. The intent of this experiment is to observe behaviour at velocities at the impact that resulted in a crash in the original simulation scenario and see if the additional altitude helps the controller avoid a crash.

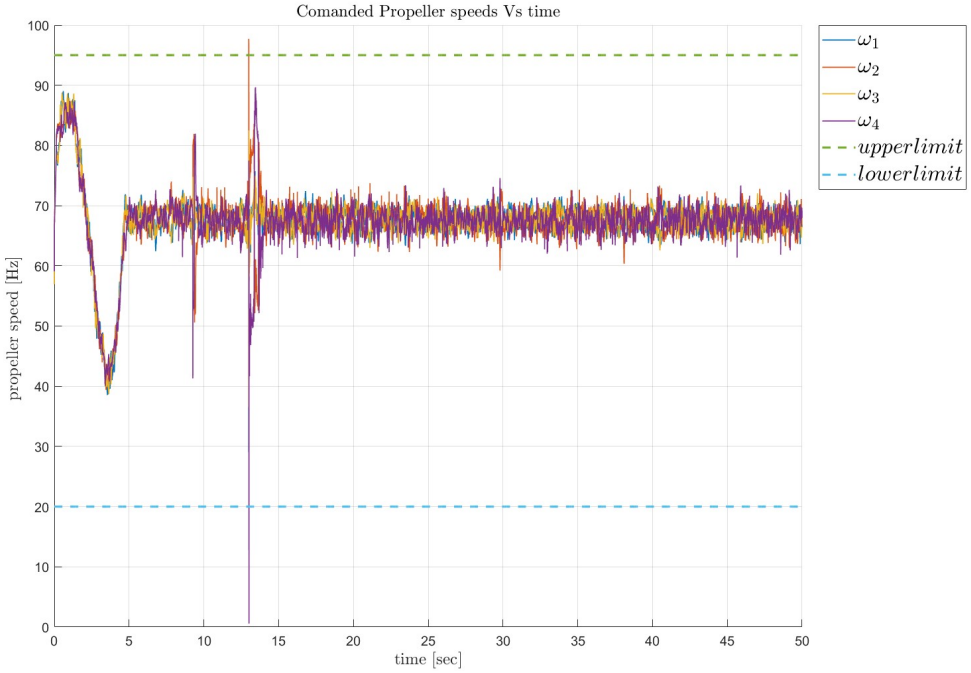
In order to isolate the effect of a high-altitude impact event, the effect of the input saturation should be removed. It is easy to tweak saturation limits in the Simulink scheme. In particular, the motor velocity limits are removed. It is to be noted that this exercise is only valuable from a theoretical point of view. In reality, any motor-driven system will inevitably have physical limits which drive the system toward saturation. The limits depend on the characteristics of the components used in the system. Therefore, by removing the limits in this case, the experiment indirectly presents generalized results about the motor specifications required to recover from an impact event of similar characteristics as the one designed in this simulation.

The simulations are repeated with the desired velocity at impact (2.5 m/s) and 4 m/s. In the first case, it is observed that the quadrotor now undergoes minimal perturbation and can successfully complete the trajectory. The behaviour is effectively captured in the input plot shown in Figures 5.2 and 5.3. In the second case, the quadrotor is seen to be affected by the impact event and falls almost 12 m before recovering and continuing on the desired trajectory.

From the above set of simulations, the two main factors contributing to the post-impact behaviour are confirmed to be the input saturation limits and the altitude of the impact event. This is in line with the earlier hypothesis and therefore offers validity to the results and defines the scope of this work, namely studying high-impact tasks with input saturations and low-altitude scenarios. The role of the impact altitude is almost hidden by the input saturation. In its influence, the controller saturates before it can effectively manage the perturbation caused by the event. By removing this limitation the controller can now recover almost immediately from the impact event at the lower desired velocities at impact. For higher velocities at impact, the controller uses the additional space granted by the high altitude of the impact for recovery therefore proving that the impact altitude is significant to the scope of this work.

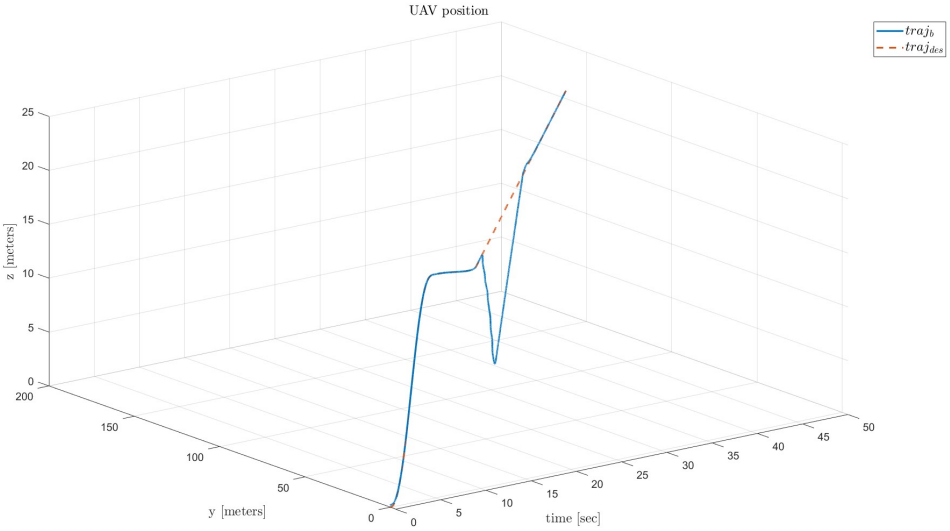


(a) Trajectory along y and z axes

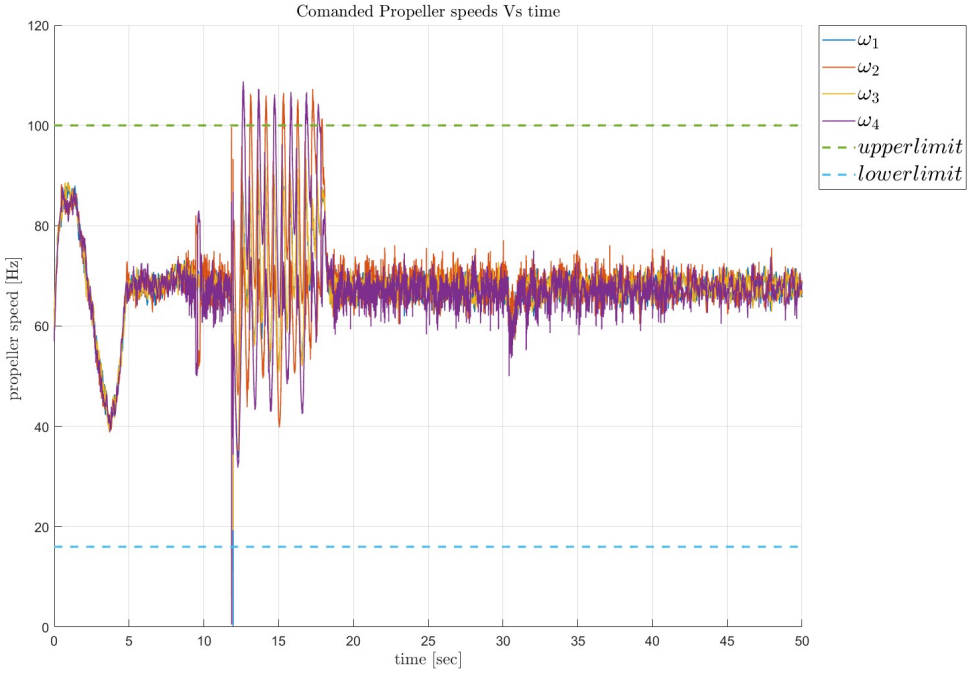


(b) Commanded propeller velocity

Figure 5.2: Behaviour for velocity at impact of 2.5 m/s. The dotted lines in (b) refer to the saturation limits of the given quadrotor. At the moment of impact (13 s) the commanded propeller speeds exceed the limits momentarily. The desired trajectory is tracked at lower velocities at impact when the input saturation is switched off



(a) Trajectory along y and z axes

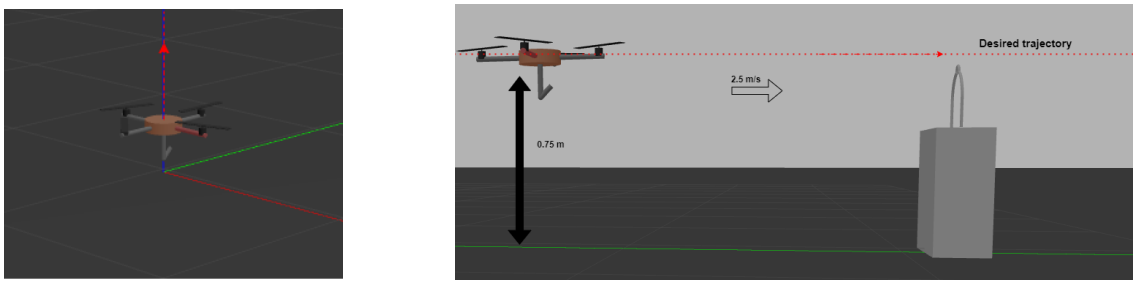


(b) Commanded propeller velocity

Figure 5.3: Behaviour for velocity at impact of 4 m/s. The dotted lines in (b) refer to the saturation limits of the given quadrotor. The quadrotor takes longer to recover and the commanded propeller speeds exceed the saturation limits multiple times in this duration. In (a) it can be observed that the quadrotor falls 12 m before recovering and following the desired trajectory

5.2 Numerical Validation of the Control Strategy

The impact-aware control strategy developed in this work is validated through numerical simulations on the three scenarios selected in Section 3.3. In each case, the quadrotor is run with both the geometric controller described in Section 2.1 and the modified controller as described in Chapter 4. As described in Section 3.3, the reference trajectory requires the quadrotor to perform uniform linear motion along the y-axis, while maintaining a constant altitude. The reference trajectory further requires the quadrotor to continue along the desired trajectory even after the impact event. Figure 5.4 shows two snapshots illustrating the sequence of events in a given validation test irrespective of the chosen scenario (or object to be impacted).



(a) (b)

Figure 5.4: Snapshots of a representative impact experiment. (a) initial configuration. (b) quadrotor follows the reference trajectory. The dotted lines show the desired reference trajectory.

Figures 5.5 and 5.6 show the relevant states of the quadrotor when the Loop simulation scenario is run with the geometric and modified controllers respectively for a desired velocity at the impact of 2.5 m/s. In both cases at 13 s, when the impact occurs, the commanded propeller speeds experience a spike in response to the perturbance. In the case of the regular geometric controller, the propeller speeds soon reach saturation levels and soon after by observing the z-pose, it can be seen that the quadrotor has crashed. The modified controller on the other hand reacts much better to the perturbation. From the commanded propeller speeds in Figure 5.6 it is clear that the perturbation is seen as a disturbance that the controller can effectively mitigate. Additionally, the re-entrance strategy is observed in the y-velocity plot of the modified controller. Figure 5.7 shows the relevant states of the quadrotor when the Loop simulation scenario is run with the modified controller. The illustrated scenario remains the same as in Figure 5.6 except for not implementing the re-entrance strategy. Here the desired velocity at impact is low enough that the modified controller does not deviate much from the desired trajectory. Therefore when tracking control is turned back on, the controller can safely bring the quadrotor back on the desired trajectory. However, it is observed that the settling time is slightly higher here. The importance of the entrance strategy is more prominent in other simulation scenarios with higher desired velocities at impact. The interested reader can refer to the Supplementary material.

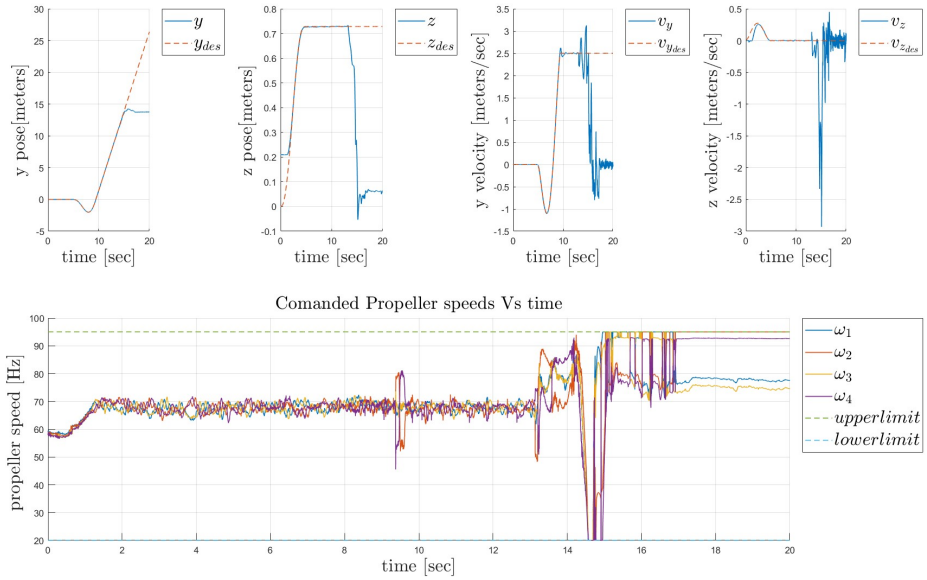


Figure 5.5: Quadrotor states when running the loop scenario with the geometric controller. The desired velocity at impact here is 2.5 m/s. Clockwise from top left: 1. pose along y-axis 2. pose along z-axis 3. velocity along y-axis 4. velocity along z-axis 5. system input (commanded propeller speeds)

In order to fully evaluate the performance of the modified controller. The three selected simulation scenarios are tested on an increasing range of desired velocities at impact until the system starts failing again. Table 5.1 provides a summary of the results obtained from the tests [1]. In all three scenarios, the modified controller improves the range of desired velocities at impact so that the system recovers from the impact. The modified controller is most effective for simulations run on the Loop scenario. This seemingly unintuitive result stems from the fact that the relative pendulum-like motion of the Loop aids in the recovery albeit increasing the resultant net torque experienced at the CoM of the quadrotor. The swing motion has a corrective effect on the net tilt experienced by the quadrotor during the post-impact period. With the regular geometric controller, the increased net torque propels the controller towards saturation. The modified controller prioritizes maintaining altitude. Therefore, the corrective swinging motion plays a bigger role here. The relative motion also serves as the most likely explanation for why the quadrotor conditionally recovers from the perturbation even with the regular controller at 3.5 m/s. It presents a unique instance where the corrective effect of the relative motion matches up with the forward tilt produced due to the perturbation. The H scenario is specifically designed to avoid the relative motion and therefore behaves closer to the block scenario where the effects of the impact are isolated from the carrying task.

[1] Results are provided for perusal in the Supplementary material

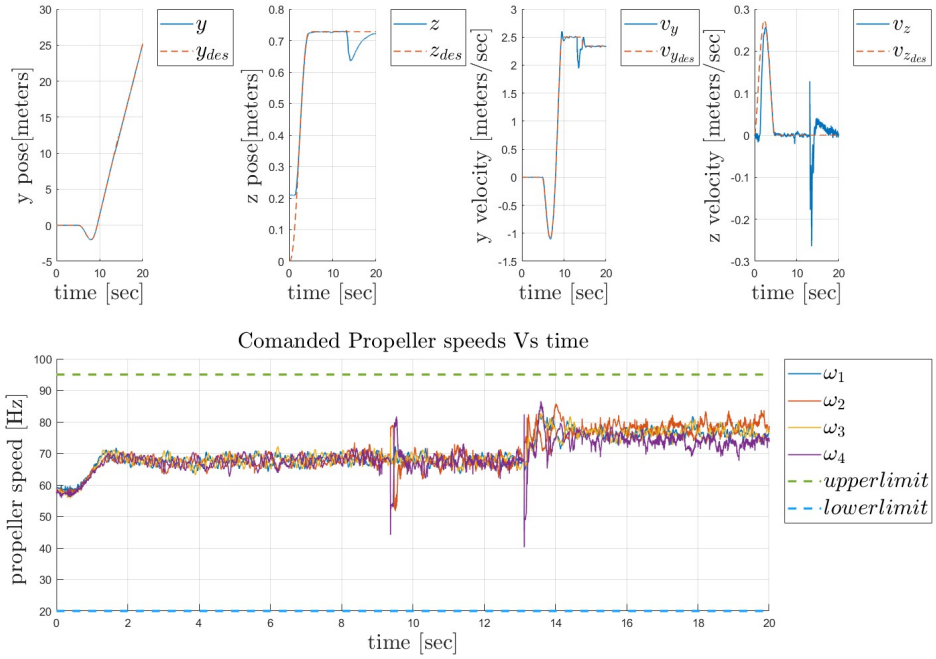


Figure 5.6: Quadrotor states when running the loop scenario with the proposed modified controller. The desired velocity at impact here is 2.5 m/s Clockwise from top left: 1. pose along y-axis 2. pose along z-axis 3. velocity along y-axis 4. velocity along z-axis 5. system input (commanded propeller speeds)

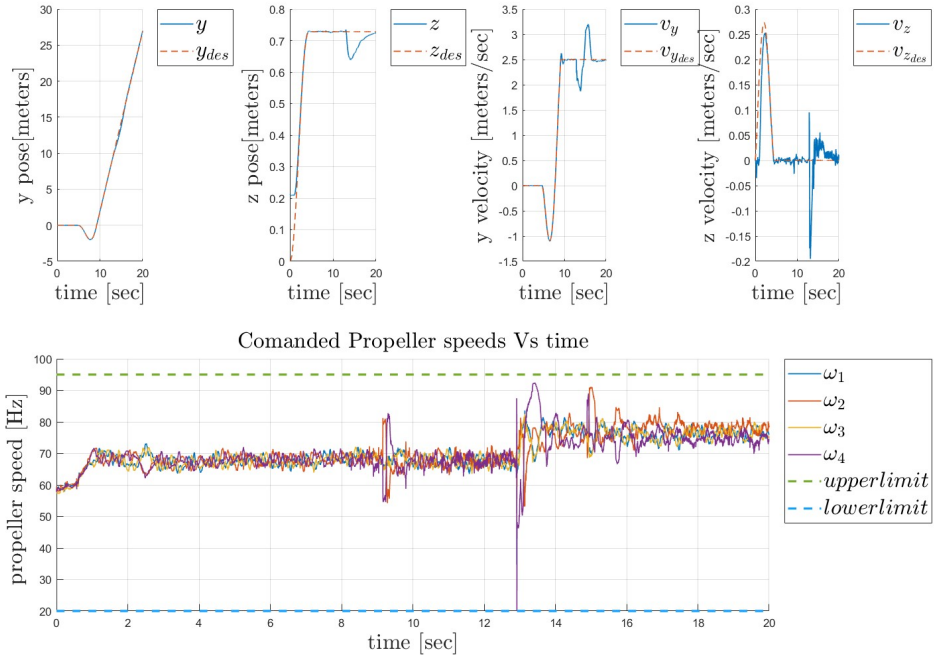


Figure 5.7: Quadrotor states when running the loop scenario with the proposed modified controller. However, the reentrance strategy is not implemented. The desired velocity at impact here is 2.5 m/s. Clockwise from top left: 1. pose along y-axis 2. pose along z-axis 3. velocity along y-axis 4. velocity along z-axis 5. system input (commanded propeller speeds)

5.3 Basin of Attraction

This section attempts to decipher the boundary that defines whether a perturbation is seen as a disturbance or an impact event by the given geometric controller. A disturbance in this context is defined as a perturbation that the controller can act upon and recover from without any external intervention. In other words, the discussion seeks to identify a basin of attraction that acts as a safe region where the controller acts as intended. Further, it dwells on how the modified controller, introduced above, affects this basin of attraction.

Once again scaled tracking errors provide a suitable starting point. The maximum post-impact error values act as a test condition to determine this boundary. The region can be visualized in a 2-dimensional sense. From the discussion in Section 4.3, the three main error signals that play a role in shaping the post-impact behaviour are identified as the velocity tracking errors along the y and z-axes and the angular velocity tracking error about the x-axis. Therefore, the translational and attitude-velocity tracking errors collectively serve as good test variables. Alternatively, the translational and attitude-position tracking errors also provide some interesting insights. To capture their collective effect the Euclidean norm is taken. Thus the final test variables are defined as:

$$E_{v_{\text{trans}}} = \max(\|K_{v_i} e_{v_i}(t)\|) \ \forall t > t_{\text{impact}} \text{where } i \in x, y, z \tag{5.1}$$

$$E_{v_{\text{rot}}} = \max(\|K_{\Omega_i} e_{\Omega_i}(t)\|) \ \forall t > t_{\text{impact}} \text{where } i \in x, y, z \tag{5.2}$$

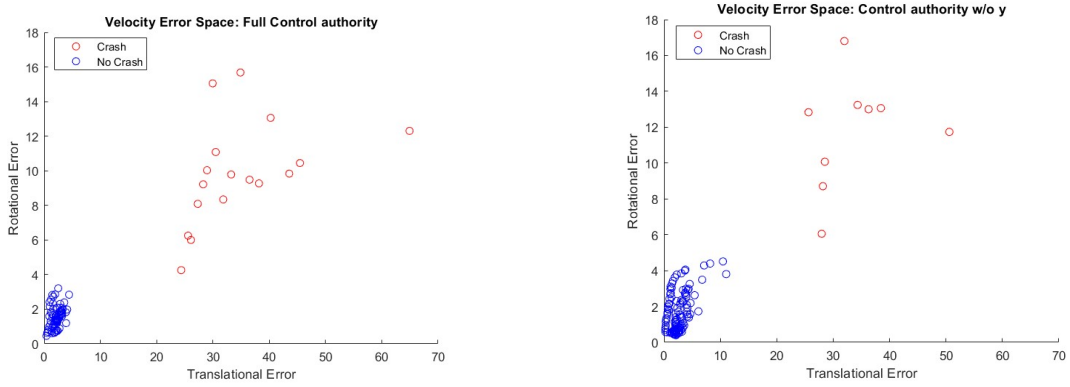
Scenario / Velocities	Loop		H		Block	
Controller	Regular	Modified	Regular	Modified	Regular	Modified
2.5 m/s	✗	✓	✓	✓	✗	✓
3.0 m/s	✗	✓	✗	✓	✗	✓
3.5 m/s	(✓)	✓	✗	(✓)	✗	(✓)
4.0 m/s	✗	✓	✗	✗	✗	✗
4.5 m/s	✗	✓	✗	✗	✗	✗
5.0 m/s	✗	(✓)	✗	✗	✗	✗

Table 5.1: Performance of the modified controller across various desired velocities at impact. ✓ signifies that the controller successfully recovers at the given desired velocity, ✗ signifies a failure and (✓) signifies that the controller conditionally recovers subject to the noise. The noise is further discussed in Appendix D.

$$E_{p_{\text{trans}}} = \max(\|K_{p_i} e_{p_i}(t)\|) \ \forall t > t_{\text{impact}} \text{where } i \in x, y, z \tag{5.3}$$

$$E_{p_{\text{rot}}} = \max(\|K_{R_i} e_{R_i}(t)\|) \ \forall t > t_{\text{impact}} \text{where } i \in x, y, z \tag{5.4}$$

Here, K_{v_i},K_{Ω_i}, K_{p_i}, and K_{R_i} are the tracking controller gains. e_{v_i}, e_{Ω_i}, e_{p_i}, e_{R_i} are the translational and rotational tracking errors as discussed in Section 2.1. The process further requires the creation of a dataset from multiple simulation runs collecting the above variables and classifying them on the basis of the occurrence of a crash. The data was collected across the three relevant simulation scenarios namely, Loop, H, and Block and for velocities at impact set between 0.3 m/s and 5 m/s at increments of 0.1 m/s using both the regular and modified controllers. The resultant dataset forms two subspaces of interest, namely the velocity error space where the translational velocity error test variable, $E_{v_{\text{trans}}}$ is plotted against the rotational velocity error test variable, $E_{v_{\text{rot}}}$. The position error space is formed by plotting the translational position error test variable, $E_{p_{\text{trans}}}$ against the rotational position error test variable, $E_{R_{\text{rot}}}$. The resultant plots are provided in Figures 5.8 and 5.9.

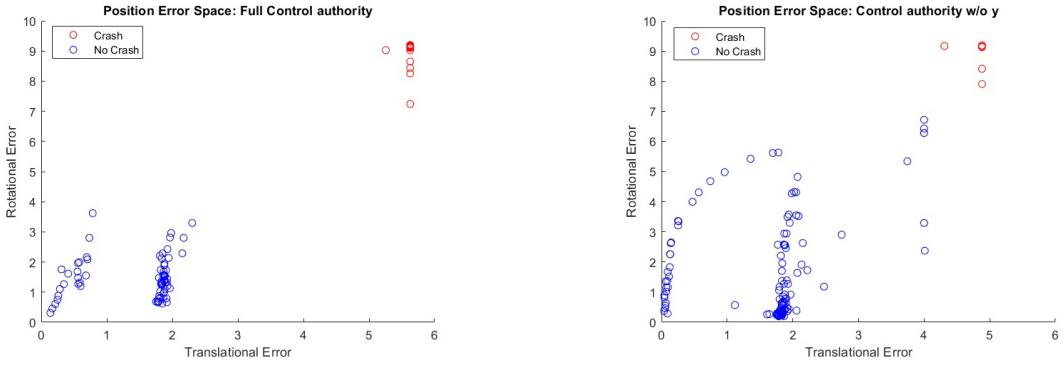


(a) Geometric Controller

(b) Modified Controller

Figure 5.8: Velocity Error Space

From the figures, it can be observed that a clear boundary exists between tracking errors resulting from perturbations that are seen as a disturbance and ones that result in crashes. Velocity tracking errors from the safe trials form a cluster about the origin which points to the existence of a clear basin of attraction. The modified controller is observed to extend this basin further. Additional data may better define the structure of this basin. In addition, the trials that see the perturbation as a disturbance form two distinct regions in the position error space. The most likely explanation for this is the two types of tasks that the quadrotor performs post-impact. In the Loop and H scenarios, the quadrotor is carrying an additional load post-impact unlike in the Block scenario.



(a) Geometric Controller

(b) Modified Controller

Figure 5.9: Position Error Space

5.4 Impact Invariant Subspace

[14] introduces a novel strategy to deal with uncertainties that arise during an impact event. In particular, it attempts to address the sensitivity in controllers to poorly defined references, tracking jumps in tracking error, and large contact forces that impair state estimation that arise from an impact event. [42] highlighted that the angular momentum remains invariant to the impact event about a given contact point. [14] generalizes this feature towards a subspace of desired tracking goals that remain invariant in the light of an impact event and coined it the impact-invariant subspace. The premise of the problem in [14] is similar enough to explore the avenue of the impact-invariant subspace in the context of impact-aware aerial robots. It is interesting to see how well the conclusions in [14] hold up for an underactuated system in an impact event. Some of the key differences here are from the fact that all impact-variant tracking errors cannot be eliminated, since some impact-invariant tracking errors may be interdependent on them.

As a means to compare the results of the two works, the impact-variant subspace of the given scenario is computed and visualized. The impact-invariant subspace is the null space of the impact-variant tracking errors. Both works assume a rigid body contact model for the impacts. This implies that the system does not allow for any deformations and impacts are resolved instantaneously. Given that the configurations remain constant over the period of the impact event, the impact variant tracking errors for a given contact impulse Λ are given as:

$$M(\dot{q}^{+} - \dot{q}^{-}) = {H_E}^{T}(R_R)\Lambda \tag{5.5}$$

where $\dot{q}^{+}$ and $\dot{q}^{-}$ are pre- and post-impact velocities and Λ is the impulse seen during the impact event. $H_E^{R^T}$ is the Jacobian matrix defined as follows [25]:

$$w_R = H_E^T(R_R)w_E,\ H_E(R_R) = \begin{bmatrix} I_3 & -[R_R p_E^R]_\times \\ O_3 & I_3 \end{bmatrix} \tag{5.6}$$

w_E is the wrench exerted by the environment on the tooltip and w_R is its effects as viewed from the CoM of the aerial robot.

Given $P(q) \in \mathbb{R}^{(n_q - n_c)\times n_q}$ is a basis for the impact-invariant subspace:

$$PM^{-1}H_E^T(R_R)\Lambda = 0 = P(\dot{q}^+ - \dot{q}^-) \tag{5.7}$$

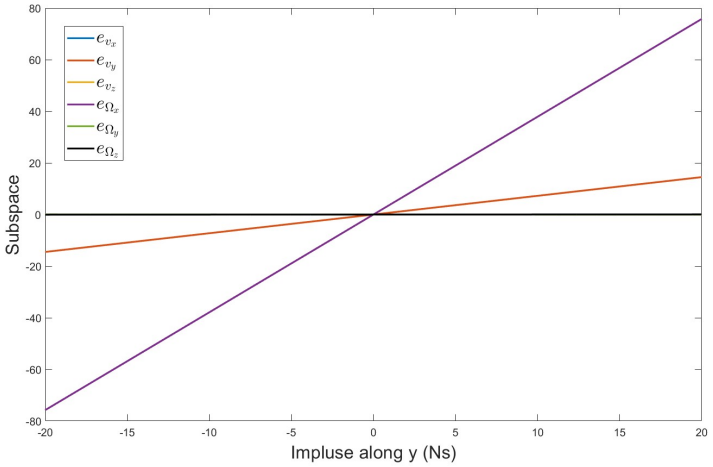


Figure 5.10: The Impact variant subspace

Figure 5.10 plots the theoretical difference between post- and ante-impact velocities as calculated using Equation 5.5 for an arbitrary contact impulse ranging from -20 Ns and 20 Ns. It shows that impact-variant subspace is composed mainly of translational velocity tracking errors [2] along the y-axis, and angular velocity tracking errors about the x-axis. Although not apparent in the graph, impact-variant subspace has a smaller effect from the angular velocity tracking errors about the y and z axes. In Section 4.1 the main factors contributing to the post-impact behaviour were determined to be the translational-velocity tracking errors along the y and z-axes and the angular velocity tracking error about the x-axis. The difference occurs from the underactuation of the system. It is established that the attitude tracking part of the controller is interdependent on the translational trajectory tracking part. Further, due to this interdependence, the final solution only avoids tracking the translational velocity tracking error along the y-axis and cannot avoid the other components identified. This result provides theoretical validation for the modified controller proposed in this work and illustrates the challenges involved in adopting the solution proposed in [14] on an under-actuated robotic system with a free-floating base.

5.5 Summary

This chapter validates the modified controller through numerical simulations. First, the hypotheses forming the basis of the proposed control are tested using the high altitude experiments. The results confirm the hypotheses that input saturation and low altitude are primary factors that

[2] The theoretical difference between post- and ante-impact velocities are equal to the velocity tracking errors due to the uniform linear motion trajectory opted here.

shape the observed post-impact behaviour. With a well-defined scope established, the proposed controller is now tested through numerical simulation with the three selected scenarios namely: Loop, H and Block. The simulation scenarios are run with both the regular geometric controller and the proposed modified controller across a range of desired velocities at impact and the results are compared. In all three scenarios, the modified controller extends the maximum desired velocity at impact before the system crashes. However, the proposed controller is most effective in the Loop scenario. This is due to the corrective effect the relative motion between the Loop and the hook has on the post-impact behaviour. Next, a dataset is created using tracking velocity errors at different velocities at impact. A clear basin of attraction is identified in this error space which distinguishes trials where the impact is seen as a simple disturbance and ones that result in a crash. In addition, the basin of attraction extends with the modified controller. Finally, the impact-variant subspace (as introduced in [14]) of the given problem is plotted. This endeavour helps in understanding some of the challenges involved in adopting pre-existing solutions of the dynamic contact transition problem to underactuated aerial robotic systems.

Chapter 6

Conclusions and Recommendations

The problem of contact transition has garnered a lot of interest owing to the development of robotics hardware and an extension in the capabilities of modern robots. In most cases, contact is still established at near-zero velocities or suffers from a finite number of bounces normal to the surface of contact when contact is established at non-zero velocities. The dynamic contact transition problem attempts to establish stable contact at non-zero velocities. It offers a number of advantages in terms of increased task execution efficiency, reduced energy consumption and increased repertoire of tasks performed by a given robot. So far, dynamic contact transition has been applied to manipulators performing picking and place tasks and humanoids performing running/jumping tasks. The problem studied encompasses both single point-of-contact impact tasks as well as simultaneous impacts. This research aims to extend the study of dynamic contact transition to aerial robotics. The following section evaluates the research undertaken in this work against the research questions formulated in Section 1.3. Following which recommendations for future research are discussed.

6.1 Conclusions

Is an impact-aware approach interestingly motivated by an application in aerial robotics?

Yes, it is interesting to apply the impact-aware approach to aerial robots as motivated in Chapter 3. The recent years have seen a rise in the number of use cases where aerial robots are employed. This should come as no surprise as flying is often beneficial in accessing remote and dangerous environments. However, most systems deployed today are limited by their battery limitations severely cutting done their flight times. Additionally, the unique set of problems associated with manipulation with a floating base limits the total force produced during interaction tasks involving the exertion of a certain amount of force. Such problems are addressed through techniques like force vectoring. However, this requires the use of additional rotors or employing fully actuated systems which greatly increase the cost and energy consumed. Exploiting dynamic contact transition in this context can help achieve the same tasks faster and utilize simple systems like the quadrotors bringing down the overall resource cost.

Applying dynamic contact transition or the impact-aware approach is a special case of aerial manipulation. In the long run, this approach can greatly increase the repertoire of tasks performed through aerial manipulation and improve the efficiency of pre-existing tasks. Some applications of special interest include swooping as seen in birds which can be used for picking up objects or collecting samples from non-solid surfaces, intercepting a target from mid-air, and force application

on a target surface. Finally, the most important result motivating an impact-aware approach is the fact that the system fails to recover from the perturbation caused by impacts at velocities greater than 2.5 m/s. The controller used here is a typical choice for aerial robots and therefore illustrates the current state of the art. This work focuses on the use case of swooping and builds a test bed simulation scenario around this application.

What distinguishes the aerial robots in this scenario from other robotic systems, i.e., which are the area-specific challenges that arise when applying impact-aware control methods to aerial robots?

Several key differences distinguish aerial robotics from other robotic systems. A few of these stand out when applying the impact-aware approach to aerial robots.

Floating Base and Input Saturation

One of the main factors affecting the post-impact behaviour was input saturation. Saturation limits are set by the design specifications of the motors used in a given UAV. The operational speed range of the motors sets limits on the total amount of thrust that a vehicle can produce. Although it may vary between vehicles, it is a common trait in all aerial robots regardless. The saturation limits in turn limit the amount of control authority available for different tasks. The floating base leads to coupled dynamics of the tool and quadrotor base. This is especially critical here since at the time of the impact event, unlike in other robotics systems, the aerial robot has to balance itself in addition to performing the interaction task. Without efficient distribution of control authority, the system can quickly reach its saturation limits and fail.

Low Altitude

A low-altitude impact event is a feature specific to the selected use case. However, it highlights certain key characteristics of extending the impact-aware approach to aerial robots. By setting the impact event to occur at low altitudes, an additional constraint is set on the system. During the impact event, the aerial robot cannot afford to lose any altitude while trying to recover from the disturbance caused by the impact. This puts additional strain on the control authority and can cause input saturation if not properly handled.

Underactuation

Underactuation is a challenge common to any problem involving the control of a quadrotor. Here, it sets additional constraints on the control strategies employed. The quadrotor tilts forward along the direction of motion in response to the impact. However, due to the underactuation, the tilt cannot be directly addressed. Further, simply correcting the tilt would have inadvertent consequences on the other tracking errors. The effect of the underactuation is clearly seen in the impact-variant subspace.

Design Limitations on Target Object

It is observed that the design of the target object affects the post-impact behaviour. This work tested two design variations of basket ear modelled as the target object for the chosen scenarios. Designing the point of impact on the target object close to its CoM reduces the total impulse seen at the quadrotor's CoM which in turn affects the post-impact behaviour. Further, it is noticed that the relative motion between the hook and the object affects the post-impact behaviour. These observations imply that the design of the target object is crucial to shaping the post-impact behaviour and in turn the solution.

What constitutes a suitable solution to control an aerial robot subject to impacts?

There are a number of possible approaches to controlling an aerial robot subject to impacts as discussed in Chapter 4. The solution opted for here shapes the reference objectives of the controller and performs well for the chosen problem. Key features from the swooping motion are translated into a trajectory-tracking task for a quadrotor. The resultant task requires the quadrotor to execute a uniform linear motion at a pre-determined desired velocity. An object is placed in the path that it encounters once it has achieved the desired velocity leading to the impact event. Three different simulation scenarios are selected based on the design of the object, namely the Loop, H and Block. In two of the selected scenarios (Loop and H), the quadrotor picks up the object during the impact event. The Block scenario acts as a control case to gain insights into the post-impact behaviour alone. When using the regular geometric controller, the selected scenarios start failing from a desired velocity of 2.5 m/s. The main causes are identified as input saturation and low altitude of the impact event.

The aim of this work is to extend the range of desired velocities at which the quadrotor can perform the trajectory-tracking task. The proposed control scheme draws inspiration from the Interaction Planning Problem. It proposes modifications to the regular geometric controller. The tracking errors that have the most influence on the post-impact behaviour are identified and a reflex reaction is designed. The controller enters into the reflex reaction mode immediately after an impact event is detected. The activation signal for the reflex reaction is chosen as the rate of the estimated external wrench. Lastly, a re-entrance strategy is proposed that ensures safe re-entry into the trajectory-tracking task.

The proposed controller is validated through numerical simulations. It is found to increase the range of desired velocities up to 5 m/s. In addition, an error space is introduced where a basin of attraction is identified that distinguishes between instances where the quadrotor perceives a perturbation from the impact event as a disturbance or instances that lead to a crash. Hypotheses about the factors influencing the post-impact behaviour are tested through a set of high-altitude simulations. The impact variant subspace of this problem provides some insights into the challenges involved with extending pre-existing dynamic contact transition control techniques to underactuated aerial robots.

6.2 Recommendations

The contributions summarized in the previous section satisfy the overall goal of this thesis. This work introduces a new research area coined as impact-aware aerial robotics. This field is explored from the limited scope provided by the purview of a Master's Thesis Assignment and provides promising results. The work here opens up several interesting avenues for future research. These topics of interest are formulated into a list of recommendations for future research below:

Experimental Validation

The immediate next step would be to validate the proposed controller through physical experiments. There is an inherent risk with this endeavour due to the highly dynamic task at hand. One approach would be to conduct experiments in an environment that allows for safe crashes. This could be a heavily padded indoor environment that allows for safe crashes. Another approach could be to design a specialized quadrotor using a soft or modular frame.

In-depth Study of Post-Impact Behaviour

An alternate line of inquiry would be an in-depth study of the Post-Impact Behaviour. A brief study was conducted in this work to understand the primary factors affecting this behaviour. However, there are a lot of features of the behaviour that might be worth looking into. A few of these points have been highlighted in Appendix C. This may open up new avenues for addressing the dynamic contact interaction problem in aerial robots.

Alternate Activation Signal for the Re-entrance Strategy

The rate of external torque estimate is used here as the activation signal for the re-entrance strategy. Ideally, the external wrench estimate signal should only measure the source of an interaction with the external environment. This would mean that the signal can only effectively measure the impact event but not the quadrotor's reaction. The choice to use this signal as the activation signal for the re-entrance strategy was strictly empirical. It is hypothesised that the wrench estimate signal measures the reaction because the addition of the manipulator changes the location of the CoM of the quadrotor. Therefore an alternate signal would be more generalisable.

Comparison with other control schemes

As discussed earlier, input saturation is a fairly common problem associated with the control of aerial robots. This work explores one solution for controlling aerial robots subject to impacts. It would therefore be interesting to explore how other solutions dealing with input saturation would fare for the given problem. Control techniques like adaptive backstepping control or nested saturation control using an admittance control external loop could better adapt to the post-impact behaviour. Another possible solution is gain scheduling. Here the controller would have to be tuned specially for the impact event and briefly switch gain upon detecting one. The solution proposed in this work focussed on modifying the existing geometric controller. However, it is interesting to compare different control schemes.

Extending existent impact-aware control techniques

The discussion regarding the impact-invariant subspace opens up another interesting avenue of future research. A number of control techniques have been introduced in recent years for the dynamic contact transition problem. These control techniques mainly cater to robotic manipulators or humanoids. Extending these techniques to aerial robots presents a number of challenges as listed earlier. Nevertheless, it is valuable to study such extensions since it opens up a link between impact-aware aerial robotics and general impact-aware robotics.

Defining the basin of attraction

The basin of attraction with the current dataset looks like a cluster with no definite form. It would be interesting to explore if the basin of attraction has a definite form. This can be done by extending the dataset further through simulation trials for desired velocities at impact over smaller intervals. A definite form would provide more insights into the boundaries between perceiving a perturbation as a disturbance or otherwise.

Exploring the approach on other aerial manipulators

The impact-aware approach developed in this work uses a passive tool that is rigidly attached to the body of the quadrotor. It is interesting to study whether the approach changes when using an active manipulator. An active manipulator increases the DoF available. It can also help decouple the effect of the impact from the quadrotor.

Exploring other use cases

Yet another possible avenue for further research is to explore other use cases. As discussed in the earlier chapters the dynamic contact transition problem has numerous applications in aerial robotics. This work adapts the swooping behaviour seen in birds to a suitable simulation scenario for aerial robots. It is especially interesting to see if the solution proposed stays relevant across various use cases of the dynamic contact transition problem.

Appendix A

Taxonomy of Multirotor Aerial Vehicles (MAV)

Multirotor Aerial Vehicles (MAV) is the term referring to the collective set of aerial vehicles characterized by a given number of Atomic Actuation Unit (AAU) placed around its body. [43] proposes a general taxonomy to describe Multirotor Aerial Vehicles and their designs. The discussion here is restricted to a quadrotor fixed propeller MAV with collinear orientations and referred to as quadrotors from here on. This section presents a general introduction to the taxonomy characterizing quadrotors and ends with the derivation of its full allocation matrix. The terminology introduced here is used throughout this thesis unless explicitly stated otherwise.

As the name suggests, quadrotors have four Atomic Actuation Unit (AAU)s. Each propeller produces force along the z-axis of its body-fixed frame where all four local z-axes are parallel to each other. The world frame is denoted by F_W (x_W, y_W, z_W) with origin as O_W. The body-fixed frame of the MAV is denoted as F_B (x_B, y_B, z_B) with origin at O_B and coincides with the Centre of Mass of the vehicle. p and R are the position and orientation of the vehicle with respect to the world frame. Further, it is noted that $p \in \mathbb{R}^3$ and $R \in SO(3)$, where $\mathbb{R}^3$ is the three-dimensional set of real numbers. SO(3) stands for Special Orthogonal Group which is a Lie group consisting of three-dimensional square matrices such that $R^{-1} = R^T$ and have a determinant of 1. The linear velocity of the vehicle in the world frame is given by $v = \dot{p} \in \mathbb{R}^3$. The angular velocity, Ω_W^{BB} of the vehicle with respect to the world frame expressed in the body fixed frame is given as $\tilde{\Omega} = \dot{R}_W^B R_W^{TB}$ where $\tilde{\Omega} \in so(3)$ is the skew-symmetric matrix form of Ω_W and so(3) is the lie algebra of SO(3).

The body-fixed frame attached to the i^{th} propeller of the vehicle is denoted by F_{P_i} and O_{P_i} denotes its origin, which coincides with its Centre of Mass. p_i denotes position with respect to the F_B. It has a spinning rate of $w_i \in \mathbb{R}$ about its z-axis, z_{P_i}. This spinning rate produces a thrust, $f_i \in \mathbb{R}^3$ and a drag moment, $\tau_i^d \in \mathbb{R}^3$ in the following manner:

$$f_i = c_{f_i} w_i ||w_i|| z_{P_i} \tag{A.1}$$

$$\tau_i^d = \kappa_i c_{\tau_i} w_i ||w_i|| z_{P_i} \tag{A.2}$$

The spinning rate is defined as $w_i = sign(\tilde{c}_{f_i})s$ where s is the component of Ω_i along z_{P_i}. The $sign(\tilde{c}_{f_i})$ indicates whether the propeller is Clockwise (CW) or Counter Clockwise (CCW). This is determined by the direction a given propeller has to spin in order to produce a thrust in the $+z_{P_i}$ direction when observing its motion from the top view. Clockwise (CW) propellers have negative $sign(\tilde{c}_{f_i})$ and Counter Clockwise (CCW) propellers have positive sign. c_{f_i} and c_{τ_i} are positive values constants that are derived from the aerodynamic properties of the propeller. κ_i is -1 if for CCW propeller and +1 for CW propeller and follows as a direct consequence of the definition of

spinning rate. The control input is defined as $u_i = w_i||w_i||$.

The total force and moment experienced at the O_B in the body-fixed frame of the vehicle F_B is given by:

$$f_B = \sum_{i=1}^{4} c_{f_i} z_{P_i} u_i \tag{A.3}$$

$$\tau_B = \sum_{i=1}^{4} (\kappa_i c_{\tau_i} z_{P_i} + c_{f_i} p_i \times z_{P_i}) u_i \tag{A.4}$$

Assuming all propellers are identical, one can define the position of O_{P_i} with respect to F_B as:

$$p_i = R_z((i-1)\frac{\pi}{2}) \begin{bmatrix} l \\ 0 \\ 0 \end{bmatrix} \tag{A.5}$$

Here, R_z is the canonical rotation matrix about the z-axis.

Any generic UAV can be described with the following control equation:

$$\ddot{b}^d(t) = h(b^d(t), \dot{b}^d(t)) + G(b^d(t))w(t) \tag{A.6}$$

which is a non-linear version of the state space control equation

$$\dot{x} = Ax + Bu$$

where h is a function of the state variables acting as the state matrix, G is the control matrix and w(t) is the wrench space. Since in a generic UAV, we have to consider the influence of change in tilt angles of the propeller, which takes more time than the rest of the system, w(t) has to be continuous. This means that such a control system cannot do jumps in its input. This can be resolved by taking the derivative of equation A.6. The derivative of the wrench map is as follows:

$$\dot{w}(t) = \frac{d(w(u_\lambda, u_v))}{dt} = \frac{\partial w}{\partial u_\lambda, \partial u_v} \begin{bmatrix} \dot{u}_\lambda \\ \dot{u}_v \end{bmatrix} \tag{A.7}$$

where u_λ is the vector containing all u_i and u_v is the input vector related to the tilt angle of the propellers. $\frac{\partial w}{\partial u_\lambda, \partial u_v}$ is the analytical Jacobian of the wrench map and is called the full allocation matrix of a generic UAV. In this case, however, we know that the tilt angles of the quadrotor are constant. As a result, the full allocation matrix is a function of only u_λ. Using equations A.3, A.4 and A.5 we can derive the allocation matrices of these configurations such that $f_B = F_1 u$ and $\tau_B = F_2 u$, where u is the vector containing the control inputs of all propellers.

$$F_1 = c_f \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{A.8}$$

$$F_2(\gamma) = \begin{bmatrix} 0 & c_f d & 0 & -c_f d \\ -c_f d & 0 & c_f d & 0 \\ c_\tau & -c_\tau & c_\tau & -c_\tau \end{bmatrix} \tag{A.9}$$

Since two rows of the full allocation matrix are always equal to zero, the quadrotor is under-actuated with $\text{rank}(F) \leq 4$.

Appendix B

Scenario Selection

Impact-aware manipulation is a relatively new field of research and to the extent of the author's knowledge, impact-aware aerial robots as a research field is completely unexplored. There are countless avenues to explore and it is easy to get lost. As a result, it is important to set a definite scope early on in the project. This is achieved through a scenario selection process that attempts to find a simulation scenario that acts as the test base for the selected problem and the subsequent solution-seeking process.

The process is naturally iterative. The initial iterations draw inspiration from impact-aware manipulation scenarios tested out on manipulators as seen in the literature reviewed in Chapter 1 while adapting them to a use case that would be relevant for aerial robots. The later iterations then draw upon the shortcomings observed in the earlier interactions and bring about improvements until a simulation scenario is designed that defines a clear problem and offers a path toward finding a solution while showcasing at least some of the characteristic problems encountered with attempting to perform impact-aware manipulation using aerial robots

The rest of the chapter systematically describes the process behind selecting a suitable simulation scenario. Section B.1 provides an overview of the whole process and the general procedure followed for each iteration. Section B.2 describes each iteration in turn in detail. Section B.3 discusses the results from the process and analyses the iterative process as a whole.

B.1 Approach

Defining a simulation scenario involves defining an outline of the ideal result, the factors that remain constant throughout the iterative process, and the variables along with their respective boundaries.

The ideal end result of this process is a simulation that faithfully represents a dynamic contact transition situation from the context of aerial robots. Further, the scenario should exhibit behaviour that is comparable to the ones seen in dynamic contact transition tasks performed with robotic manipulators. The problem presented should be in a stage where the impact-aware techniques developed in the literature provide a starting point for the solution-seeking stage. Simultaneously, the scenario should present a use case generally encountered in applications involving aerial robotics and is therefore relevant.

A few factors of the simulation scenario remain constant throughout the iterative process. The scenario always consists of a quadrotor and tool, attached under the CoM of the main body of the quadrotor, performing a dynamic contact transition task. The task always entails one single-point-of-contact impact. No additional environmental disturbances like wind are considered. The

model of the quadrotor is also kept constant over the different iterations.

The variables factors in the simulation scenario are as follows:

- The design of the passive manipulator attached to the quadrotor
- The object to be impacted
- The trajectory undertaken by the quadrotor to perform the given task
- The use case
- The impact conditions like the points of impact on the tool and the object respectively and impact altitude.

Some amount of creative freedom is available regarding the design of these variables, however, all designs are grounded in reality and borrow their foundations from a viable real-world situation. These variables act as the building blocks towards achieving the objectives laid out while planning a given scenario.

In each iteration, the background knowledge available is analyzed and a use case is defined. The design of each of the variables defined above is determined based on the defined use case. The simulation is set up and run (often several times to account for the noise), the results are recorded and the insights are used as background knowledge for the next iteration.

B.2 The Iterative Selection Process

Throughout this project, a number of simulation experiments were conducted to isolate impact-driven behaviour in Aerial robots. The task was fairly complex owing to the different parameters affecting a given scenario. As a result, the task underwent different levels of optimization before arriving at a satisfactory test scenario. Each of these iterations is in turn examined for the key objectives they seek to achieve, how the variable factors available are designed to fit the objectives, and the main insights they provide towards shaping subsequent iterations.

B.2.1 Passive Manipulator and Block

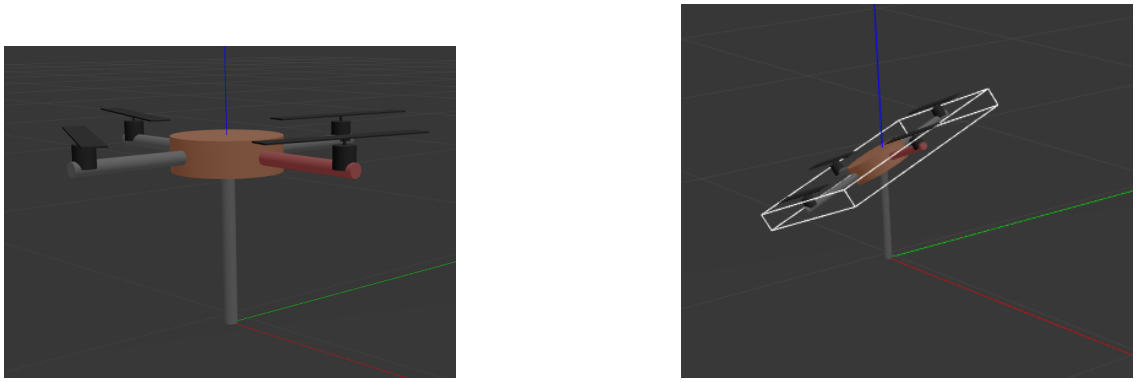
Dynamic contact transition tasks for manipulators as seen in [11] or [15] are designed such that the end effector encounters a non-elastic single-point-of-contact impact against a rigid body. This served as the template for designing the first simulation scenario. From an aerial robotics perspective, it translated into a manipulator attached to the body of the quadrotor impacting a rigid block. Without losing generality, the manipulator is simplified to a passive one-link manipulator shaped like a stick. The first interaction required some additional steps namely optimizing flight with the passive manipulator before implementing the scenario.

Flight with Passive Manipulator

The controller is initially optimized to fly just the quadrotor. When the additional manipulator is added, it changes the total mass of the system, its distribution, and moment of inertia. Therefore the controller needs to adapt to these changes. This optimization was performed in two phases. The first phase involves experimenting with the physical aspects of the manipulator namely, its mass and length. At the end of this phase, it was concluded that the stick would have a mass of 100 g and a length of 0.125 m. The mass of the stick is almost $1/10^{\text{th}}$ the mass of the quadrotor and was determined to be light enough to not disrupt the flight while supporting an impact. The length was chosen such that any impact would be far enough so as to not cause any damage

to moving parts of the quadrotor, the position (below the body) also further helped in this matter.

The second phase focused on the type of joint used to attach the manipulator to the body of the quadrotor. Since the manipulator is passive in nature, there are essentially two types of joints that can be used either a fixed joint or a ball joint[1]. The ball joint allows the quadrotor to remain unaffected by the dynamics of the manipulator. In other words, it decouples the dynamics between the quadrotor and the manipulator and hence optimizing the controller is easier. In the coupled case, the manipulator's dynamics need to be compensated by the quadrotor. Such a scenario can quickly lead to instability. One way to approach this issue is to look at the manipulator dynamics as a disturbance and improve the torque disturbance rejection of the controller. Both joints offer interesting scenarios to study. Figure B.1 shows the two designs.



(a) Coupled Case

(b) Decoupled Case

Figure B.1: First design iteration of the passive manipulator

Adding a manipulator at the bottom of the quadrotor's body makes it hard to start the flight with uniform initial conditions. This is solved by setting the quadrotor over a hollow cylinder such that the end of the manipulator is slightly above ground. The cylinder as shown in Figure B.2 acts as a launch pad that stabilizes the take-off and brings about a certain amount of regularity to the initial conditions.

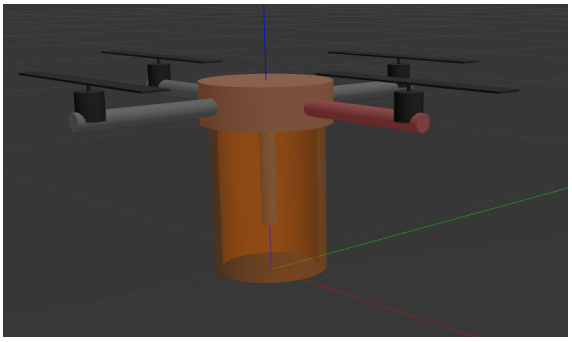


Figure B.2: Quadrotor launchpad

[1]The ball joint is later simplified to a revolute joint along the y-axis . This change does not significantly affect any results since the trajectories chosen throughout this work are two-dimensional.

Impact with Block

The object in this scenario is a simple block rigidly attached to the ground. The only aspect of importance in the block is its height. The block is high enough that the impact event occurs away from the ground. A simple trajectory is chosen where the quadrotor initially gains the required altitude and then flies toward the block. This iteration is focused on merely observing an impact event and therefore does not pay particular attention to the impact conditions. The block is placed ≈1 m from the quadrotors' starting position along the x-axis.

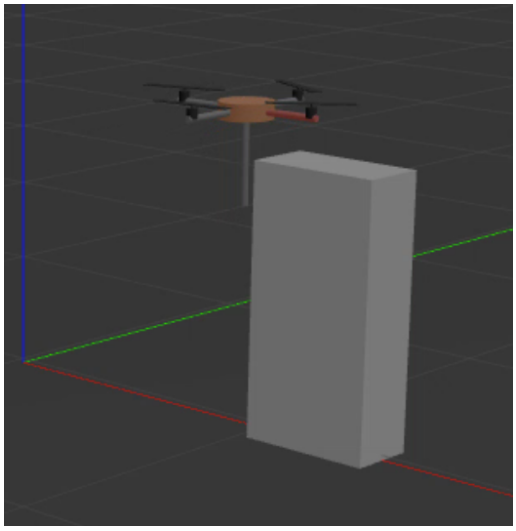


Figure B.3: Screenshot of the Impact with Block simulation scenario

Figure B.3 is a screenshot of the simulation scenario with the fixed joint manipulator case. Upon impact, the quadrotor falls along the ball joint in the decoupled case and tilts in the direction of impact in the coupled case. The velocity at impact is found to be too low to observe the effects of dynamic interaction. Additionally, it is observed that stopping the trajectory soon after the impact event does not lend enough time to observe post-impact behaviour. Lastly, due to the floating base, unlike in the manipulator, designing a non-elastic impact is not straightforward.

B.2.2 Swooping Motion

As discussed, the first iteration had a variety of flaws. Additionally, the above task was limited in its real-world applications. Any possible applications like certain inspection tasks were already accomplished more efficiently in a static manner. Owing to the limited use case of the first task, the next iteration's objective was to design a task with a relevant practical application that would greatly benefit from the dynamic nature of the interaction.

The task in this iteration resembles a swooping motion. Such a motion is helpful in situations where an object has to be picked up dynamically often from a surface that is not solid. The tool is modified to a hook with a sharp kink (Figure 3.1a). The object is now shaped like a loop which represents the ear of a basket (Figure B.4). The related scenario is designed as follows. The quadrotor spawns on the launch pad and the object is placed an arbitrary distance along the y-axis. The trajectory involves the quadrotor gaining some altitude, then swooping down (flying downwards (-z-axis) and horizontally along the y-axis), picking up the object from the ground, and flying back upwards.

Since picking up the object is a priority, the trajectory as shown in Figure B.5 is carefully

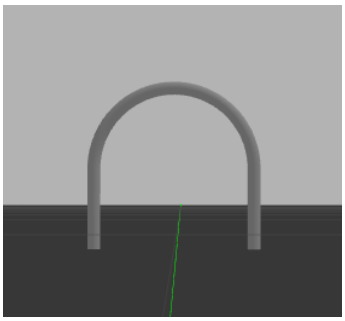


Figure B.4: First design iteration of the loop

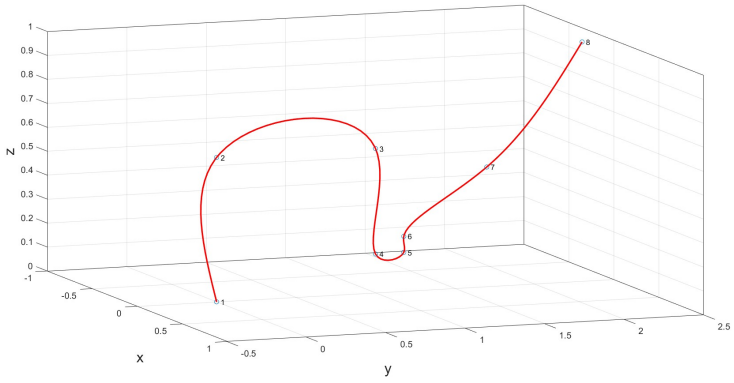


Figure B.5: Trajectory specifically designed for the use-case

designed with various via points located around the position of the object. However, this makes the impact event occur at almost zero velocity. The trajectory generation algorithm used in this iteration takes via points as inputs and creates a point-to-point trajectory between consecutive via points. Such an algorithm causes the vehicle to stop at each via point which is not desirable in this case. Due to the small contact surface (2 circular contact surfaces of radius 0.01 m), the loop can topple over easily, making the task more complicated.

B.2.3 Dynamic Trajectory

This iteration seeks to make the trajectory described in the last iteration more dynamic and make the impact occur at non-zero velocities. The use case and tool design are the same as in the previous iteration.

The objectives of this iteration are achieved by modifying the trajectory and the loop design. The trajectory generator algorithm used in the previous iteration is replaced with the Simulink Polynomial Trajectory Generator block which allows for setting desired starting velocities at a given via point leading to more dynamic executions. The resultant trajectory is shown in Figure B.6 The loop is redesigned to be wider so as to enable better access to the hook. Additionally, the hook is provided with rail-like supports that prevent the loop from toppling over easily. Figure B.7 shows the new design.

In this run the task is executed more dynamically and the design modification to the loop ensures that it is picked despite the small differences in the trajectory between multiple runs. However, a lot is happening at the same time. The drone accelerates and decelerates multiple times over the desired trajectory. The impact occurs in one such cycle and therefore the post-

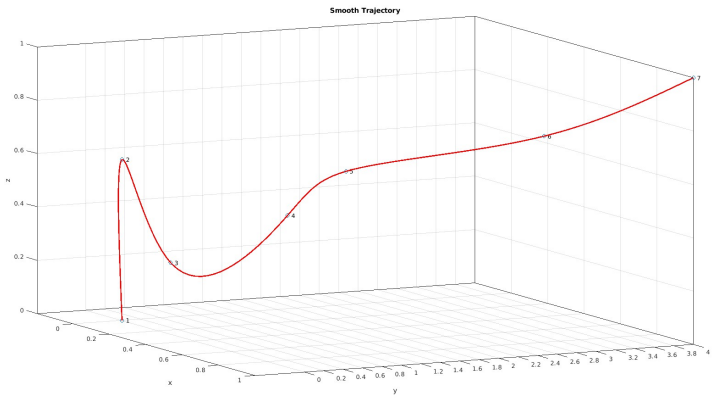


Figure B.6: Trajectory of swooping use case executed more dynamically

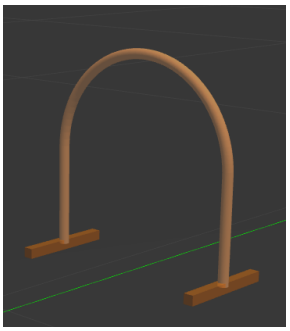


Figure B.7: Redesigned Loop with support rails

impact behaviour is also influenced by the additional quadrotor dynamics involved in tracking the desired trajectory. Figure B.8 clearly captures the dynamics through the quadrotor's velocity along the y-axis.

B.2.4 Constant Velocity Trajectory

So far in this process, the trajectory has been designed to fit a particular use case. However, observations from the last iteration suggest that focusing on the use case alone complicates the trajectory and takes the focus away from the impact event and consequently the post-impact behaviour. The objective of this iteration is to eliminate all other disturbances, particularly from the additional quadrotor dynamics required to track a complicated trajectory.

The goal here is to simplify the trajectory, around the impact event while still achieving the basic requirement, namely, picking up the object. The simplest form of motion is the rectilinear motion. By placing the loop at an elevation, it can still be picked up by a quadrotor performing uniform linear motion. The scenario then becomes as follows. The quadrotor is spawned as usual atop the launch pad while the loop is now placed at an elevation of approximately 0.75 m. The quadrotor attains the altitude such that it is at the same height as the loop, then it gains the predetermined speed and maintains the speed as it approaches the loop. The desired trajectory as shown in Figure B.9 expects the quadrotor to continue maintaining the preset speed even after the impact until the end of the simulation. The distance between the quadrotor and loop is set such that the quadrotor completely settles into the uniform linear motion trajectory before the impact event.

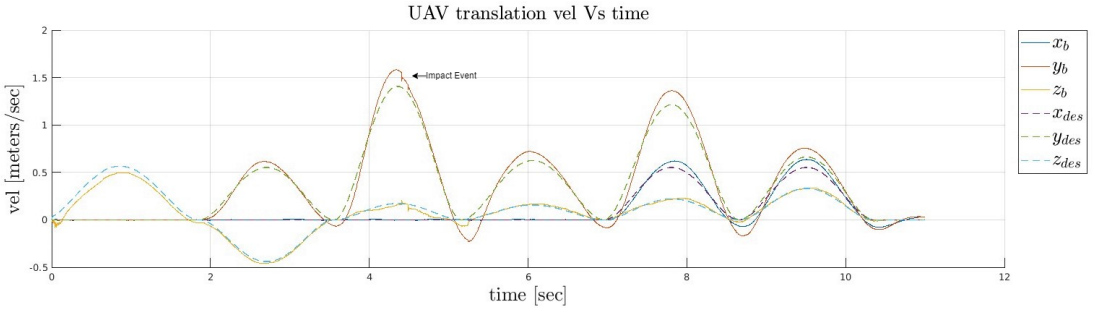


Figure B.8: Plot of the quadrotor velocity. The impact event occurs around 4.5 s. The post-impact behaviour is also influenced by the additional quadrotor dynamics involved in tracking the desired trajectory. Additionally, during the impact event, the controller is still settling into the desired trajectory

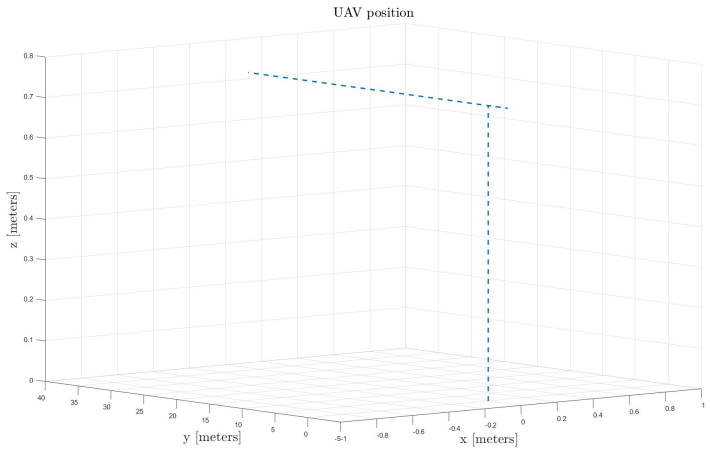


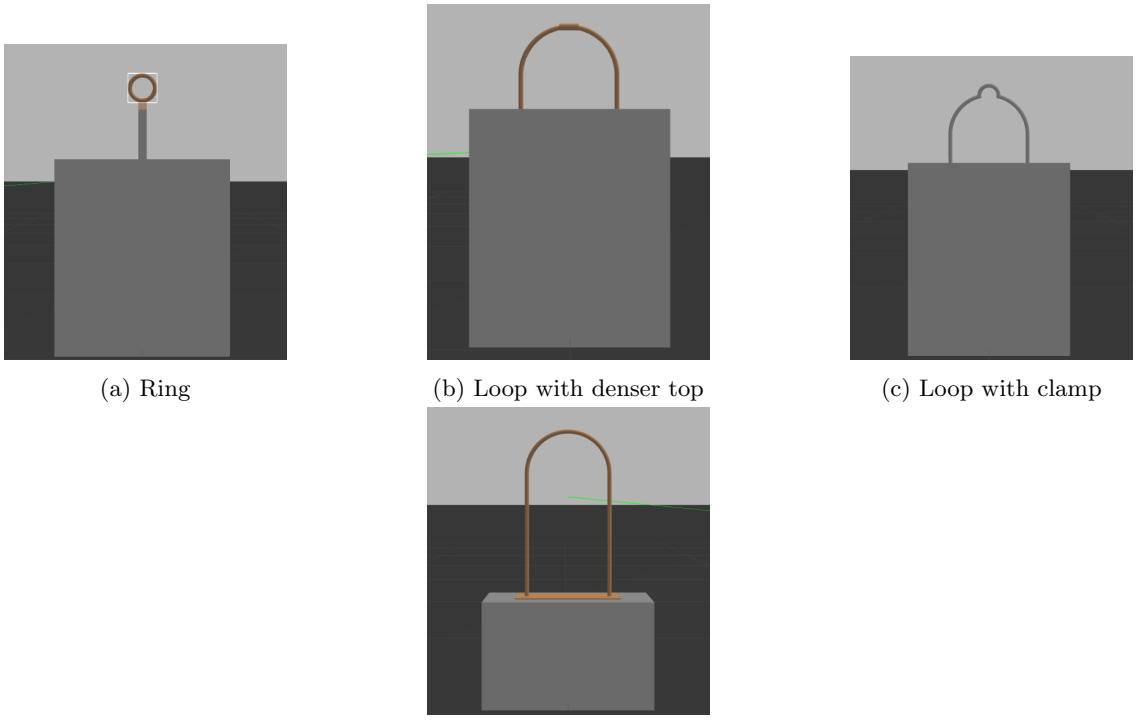
Figure B.9: The uniform rectilinear trajectory

The trajectory is now able to clearly able to distinguish the impact event. Therefore the trajectory now offers a test bed to observe the impact event. By determining a velocity for the quadrotor beforehand, the scenario now offers the additional capability to simulate an impact event at a certain desired velocity at impact. It is therefore possible to observe what velocities at impact cause a perturbation that the simple geometric controller is unable to recover from. The design of the hook and loop ensures that the pick-up occurs. However, it is observed that there is significant relative motion between the quadrotor and loop. This motion is accentuated in the decoupled case since the joint between the quadrotor body and hook makes the whole system act like a double pendulum that quickly becomes unstable. This instability has little to do with the impact event itself and therefore hinders observing isolated post-impact behavior.

B.2.5 Eliminating the Relative Motion

As seen in the previous iteration, the relative motion between the loop and the quadrotor masks the true post-impact behaviour. As a result, this iteration focuses on eliminating this relative motion. The focus is on the design of the object, a number of alternative designs are compared against a set of criteria.

Firstly, the decoupled version of the hook design is discounted from discussion for the rest of the work. The double pendulum-like motion is beyond the scope of this work and takes attention away from studying behaviour solely affected by the impact event. Secondly, the design of the object can be modified to reduce the relative motion. On close observation, the object seems to roll on the kink of the hook. The following design modifications (Figure B.10) are considered to mitigate this behaviour:



(a) Ring

(b) Loop with denser top

(c) Loop with clamp

(d) Longer Loop

Figure B.10: Design modifications evaluated for the object

- Add frictional properties to the surfaces of the contact
- Add damping properties to the surfaces of contact (Practically, this would be like adding a skin of a material with high damping properties)
- Change the object to a ring of a very small radius, such that mass is distributed evenly and concentrated close to the point of contact on the hook
- Loop with a denser top. The idea behind this design is to ensure more mass of the loop is concentrated around the point of contact
- Loop with a clamp, so that loop is somewhat fixed to the kink of the hook.
- Design a longer loop such that the force of gravity prevents the loop from rolling around.

Each of these modifications is tested through multiple simulation runs and evaluated on the basis of four criteria.

- **Practicability** helps evaluate the modification without losing sight of the original use case. The loop is supposed to represent the ear of the basket and therefore the design modifications should still represent this in principle.

- **Ease in terms of time and effort** evaluates the design modification based on the time and effort required to set them up and the ease with which pick-up occurs.
- **Generality** evaluates whether modifications restrict the use case in some manner. The results from this work should offer insights into the impacts of aerial robots in a somewhat broad sense. A hyper-specific simulation scenario that will probably never occur in real-life situations is undesirable.
- **Effectiveness**, finally, evaluates the designs on whether they in fact reduce the relative motion between the object and the quadrotor.

	Practicability	**Ease in terms of time and effort**	**Generality**	**Effectiveness**
Friction	High	Easy to make changes in simulation	Somewhat	Reduces relative motion
Damping	High	Easy	High	Reduces relative motion
Both	High	Easy in simulation	Somewhat	Reduces relative motion
Ring	Somewhat	Quite a few initial adjustments	Restricted to specific object classes	Reduces relative motion
Dense on top	High	Hard to optimize	High	Reduces relative motion
Clamp	High	Easy	Somewhat	High
Long Loop	High	Hard to optimize	Restricted to specific object classes	Requires further optimization

Table B.1: Evaluating the object's design modification against the chosen criteria

Based on the results of the evaluation presented in Table B.1, Loop with a clamp offers the best solution. It is practicable since an ear of the basket as in the originally proposed use case an be modified to include a clamp without big modifications to the use case itself. The modification was easy enough to bring about in the simulation and is similarly easy to implement practically as long as the decision is taken early on in the design stage. The design modification is a generalization to picking any object with a handle. Objects that don't fit this criteria may require additional workarounds. Finally, the clamp modification was the most effective in reducing the relative motion between the object and the quadrotor.

B.2.6 Object Center of Mass

This set of experiments does not necessarily constitute another iteration but rather a parallel path. The idea behind these experiments is the same as the last iteration; reduce (if possible eliminate) the relative motion between the loop and the hook. Rather than attempt to modify the existing object design, in this set of experiments, the object is redesigned.

The object is now shaped like the English alphabet, H, The point of impact is designed to be at the Center of Mass (CoM) of this object. By ensuring that the impulse from the impact is applied

at approximately the CoM of the object, the resultant torque about that point is zero. Therefore, theoretically, it should curb the relative motion. The object is designed as seen in Figure 3.2b. The shorter ends on the top ensure that the object does not come in contact with the quadrotor anywhere other than the intended point of contact.

The interesting observation with this set of experiments is that the system crashes a lot less often at the same velocities at impact when compared to the other experiments. It can be hypothesized that this is because post-impact behavior is formed from two sources, a torque produced at the quadrotors' CoM due to the impulse produced by the impact event and the torque originating at the CoM of the object translated to the CoM of the quadrotor. The H-shaped object curbs the second source and therefore decreases the collective effect of the impact. However, the given hypothesis is to be investigated further. The H-shaped object on the other hand isn't very practical in relation to the use case but it offers sufficient theoretical insights to be included in the final set of simulation scenarios.

B.3 Discussion

The final test bed entails the following simulation scenario. In the beginning, a quadrotor, with a hook fixed to the lower end of the mainframe, is spawned atop a hollow cylinder acting as a launch pad. The object is placed atop an elevation at a given distance calculated such that the quadrotor has stabilized itself into its desired state. The desired trajectory requires the quadrotor to gain altitude, achieve a certain translational desired velocity along the y-axis and maintain that velocity till the end of the simulation. The impact occurs at the constant velocity phase of the trajectory. Upon contact, the object gets lodged in the hook of the quadrotor and is carried for the rest of the simulation run. The two objects considered here are the clamped loop and the H-shaped object. The clamped loop aligns well with the selected use case while the H-shaped object offers some insights regarding the physics of the object. The impact altitude is chosen to be half a meter above the ground. A swooping motion usually targets low-lying targets and therefore low altitude of the impact fits well with the use case. The consequences of the chosen altitude are studied in Section 5.1.

The final test bed picks a use case that is highly relevant to aerial robots. Imitating the swooping motion can help save vital time and battery power. Thereby greatly increasing the task capacity of an aerial robot. The final test bed also faithfully represents the dynamic contact transition problem with a single-point-of-contact impact event. Appendix C studies post-impact behaviour observed with the selected test bed at different desired velocities at impact.

Appendix C

Study on Impact driven behavior of Aerial robots

The finalized simulation test bed now offers an opportunity to study post-impact behaviour. Certain traits were observed during the simulations but the focus was on finding a scenario that fit all the requirements. This Appendix is divided as follows, Section C.1 describes the post-impact behaviour observed and groups the behaviour into certain classes. Section C.2 details the different possible causes underlying the observed behaviour. Section C.3 details the latest set of simulation studies conducted and Section C.4 analyses the results and details reasons why observed behaviour may be impact-driven. Section C.5 provides possible counterarguments and points of interest that require a second look and Section C.6 lists the inferences from this study.

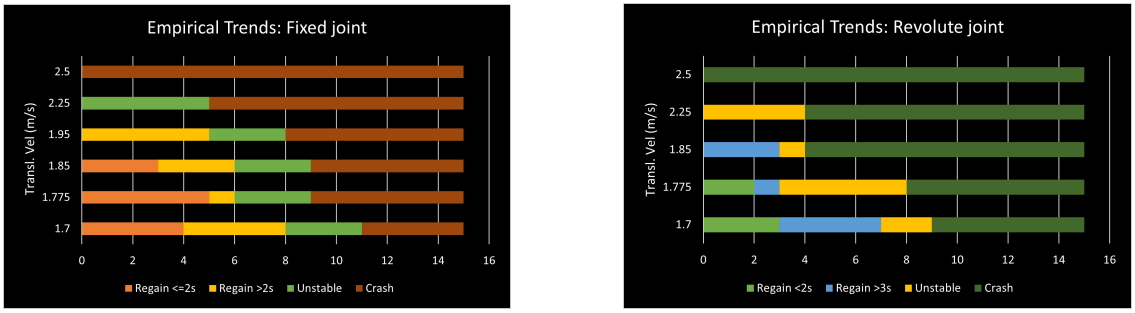
C.1 Post-Impact Behaviour

Data is collected from multiple simulation runs at velocities at impact varying from 1.5 m/s to 2.5 m/s and using the clamped loop as the object. A single simulation run at a certain velocity at impact is called a trial. Fifteen trials are conducted per velocity at impact. The subsequent post-impact behaviour observed can be classified into the following subclasses:

- **Class 1:** Recovers from the perturbation
  - **Fast Recovery:** Takes $\leq$ 2s to recover fully
  - **Slow Recovery:** Takes $>$ 2s to recover fully
- **Class 2:** Unstable but Flying
- **Class 3:** Crashes immediately after the impact event

Figure C.1 show the empirical trends observed in the post-impact behaviour for a given velocity at impact across all trials for the coupled and decoupled joints of the tool respectively. It can be observed that the incidence of crashes is higher at higher velocities at impact. At a velocity of 2.5 m/s, all trials fall under Class 3 behaviour. Between 1.5 m/s and 2.5 m/s the behavior changes between trials. Additionally, trails using the decoupled hook joint perform worse than the coupled joint case. As discussed earlier, a double pendulum is formed in this case which destabilizes the system.

Further, the simulation is run with the H-shaped object for velocities at impact $\geq$ 2.5 m/s. Here it is observed, that the trials fall under the Fast Recovery subclass of Class 1 for the velocity at impact of 2.5 m/s. At a velocity at the impact of $<$3 m/s the behaviour changes between trials. above which all trials fall under Class 3 behaviour. Therefore the range of velocities at



(a) Fixed joint manipulator

(b) Ball joint manipulator

Figure C.1: Post-Impact Behaviour: Empirical Trends

impact where the post-impact behavior recovers fully is extended when using the H-shaped object.

The rest of the work focuses on velocities at impact that result in post-impact behaviour falling under Class 3 and seeks to find ways to aid the controller toward Class 1 behaviour. As discussed in Appendix B, the primary goal of this task was to observe impact-driven behaviour and identify similarities with cases of impact in robotic manipulators. In particular, the objective sought to observe the phenomenon of peaking. Peaking occurs when, in response to an impact event, the controller aggressively counteracts the effect and in turn, leads to an exponential increase in error and subsequent failure. From observing the post-impact behaviour and quadrotor states, the following preliminary inferences were drawn:

- No significant peaking behaviour could be observed from the quadrotor's states
- The behaviour cannot be solely attributed to the impact event
- Accelerating the object with non-zero mass and inertia towards the desired speed plays a significant role in shaping the quadrotor's post-impact behaviour.
- The object's CoM and the point of contact are non-trivial in the observed behaviour.

C.2 Investigating Cause of Post-Impact Behaviour

Given these inferences, the next step is to determine the true extent the impact event plays in shaping the post-impact behaviour. It is important at this juncture to understand what event drives the behaviour observed in the finalized scenario. Therefore it is necessary to undertake a detour and further study the observed behaviour. The purpose of this study is to:

- Identify the event (or events) that primarily drives the observed behaviour
- Determine whether the behaviour is a consequence of the particular simulation scenario or inherent to aerial robots when they encounter impacts.

This study is conducted through the design of two additional simulation scenarios that isolate the impact event from the carrying event. Observations from these simulation scenarios furnish additional details on the above-drawn inferences. This section gathers and analyzes the data obtained over the course of multiple simulation runs. Through this analysis, certain characteristic behaviours and possible factors influencing this behaviour are identified. Further simulation scenarios eliminating certain factors are designed and the behaviour observed is analyzed for the characteristics identified. This analysis is then presented as proof of the dominance of certain factors over others in post-impact behaviour.

C.2.1 Possible Explanations

As described in Section 3.3, all trials exhibit certain characteristic behaviours. The scenario is composed of two main events: the quadrotor flight and the impact event. However, due to the specific use case behind the design of the test scenario, the impact event is, in fact, a combination of two sub-events. The effect of the impact itself on the quadrotor and the act of carrying an object of non-negligible mass and inertia, that undergoes sudden acceleration and deceleration. It is hard to distinguish the individual effects of these two sub-events in the post-impact behaviour of the quadrotor. As a result, it was quite unclear what caused the crash. In the rest of this section, evidence of each of these sub-events causing the crash is presented on the basis of the key characteristics identified in Section 3.3.

Impact sub-event as the cause of crash

The momentum generated due to the impact at a higher ante-impact velocity may intuitively seem like the possible cause for the crash. The impact occurs at a point below the CoM of the quadrotor, which generates a torque causing the quadrotor to tilt forward. The behaviour of the commanded propeller speeds suggests that the controller aggressively attempts to correct the error but in turn, reaches saturation and therefore crashes. The key difference here is that the quadrotor states do not behave in the manner seen during an impact in robotic manipulators. This can be explained as a consequence of the fact that unlike in manipulators, the propeller speeds are the real inputs of the system. Additionally, since the propeller speeds quickly reach saturation levels, it is hard to discern if the behaviour would be classified as peaking when provided with higher saturation limits and more space before the quadrotor hits the ground.

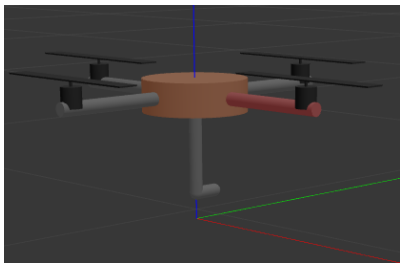
Carrying action as the cause of crash

On the other hand, the impact event also involves the addition of an object of non-negligible mass and inertia to the quadrotor assembly at a certain distance from the CoM of the quadrotor. The object is at rest ante-impact. Post-impact, the object accelerates from 0 m/s to the set velocity of the quadrotor at the instant of impact and then maintains that velocity in a relatively short span of time. Both the addition of mass (and inertia) and its relative motion (obtaining and maintaining the required velocity) are contributing factors to the perturbation observed soon after the impact event, however, their exact effect is not intuitive. It is further noticed that during the runs resulting in a crash, the quadrotor tilts back in the opposite direction in response to the forward tilt. Although this could be attributed to the aggressiveness, not unlike that observed in classical controllers involved in manipulator impact scenarios. However, since the object's dynamics are still in the process of stabilizing at this moment in the simulation, it may influence this particular behaviour. The experiments conducted, with the object modified such that impact occurs close to its CoM (H with clamp), emphasize the importance of the object's CoM in reducing instances of a crash at a given velocity at impact.

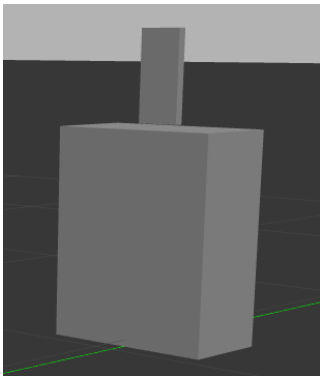
In general, the post-impact dynamics are complicated with the additional task of carrying an object. All factors discussed above play a role in the behaviour observed and it is, therefore, naive to assume that the impact sub-event is the sole or even the main contributing factor.

C.3 Additional Simulation Scenarios

Based on the findings above, two sets of additional simulation scenarios were designed. These scenarios somewhat isolate the two sub-events identified earlier.



(a) Modified hook design



(b) Block design

Figure C.2: Design changes for the 'Impact Isolation' Experiments

C.3.1 Isolating the Impact sub-event

The simulation scenario used here is similar to the scenario described earlier. The key differences are in the design of the hook and the object. Here the hook-end is fashioned at right angles with its body. Additionally, the end is designed as a sphere to ensure a single-point contact during impact. The object is designed as a thin block. The desired trajectory remains the same.

The quadrotor impacts the block at its centre and causes it to topple over while the quadrotor continues on its desired course. Thus, the experiment isolates the impact sub-event. In this case, it is safe to assume that the post-impact behaviour of the quadrotor is due to the single-point impact with the block.

C.3.2 Isolating Carrying sub-event

In this scenario, there is no impact event during the entire run of the simulation. Here, the quadrotor is carrying the hook from the beginning of the simulation. As a result, the quadrotor is spawned higher than earlier to accommodate for the loop on its hook. The loop is spawned approximately on top of the kink in the hook and settles on the hook before the start of the simulation. The launch pad is modified accordingly to accommodate the new initial conditions. The desired trajectory remains unchanged.

The primary goal of this experiment was to determine whether the controller gains needed adapting when the quadrotor flies with the loop. This experiment manages to capture certain features of the 'Carrying' sub-event. The quadrotor now flies with the object of significant mass and inertia placed away from its CoM. However, the relative acceleration observed, between the quadrotor and the object, during the carrying sub-event is not replicated in this scenario.

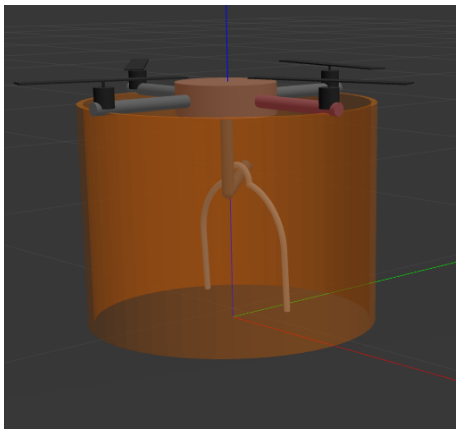


Figure C.3: Configuration for 'Flying with Loop' Experiments

C.4 Analysis

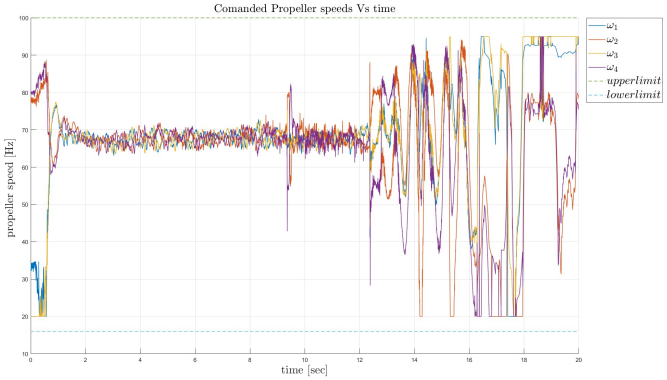
In this section, the outcomes observed from experiments described in Section C.3 are analyzed and compared with behavioural characteristics identified in Section 3.3. The section ends with inferences drawn based on these observations. It is to be noted that, unless otherwise stated, all the reported outcomes are for the velocity at impact equal to 2.5 m/s.

The anticipated outcome from the 'Impact Isolation' experiments was that the quadrotor easily recovers from the perturbation due to the impact and continues on its intended trajectory. However, the quadrotor still crashes in most instances similar to the first simulation scenario. Here, any other influence is practically eliminated since the block topples over as anticipated and provides a clear path for the quadrotor to continue on.

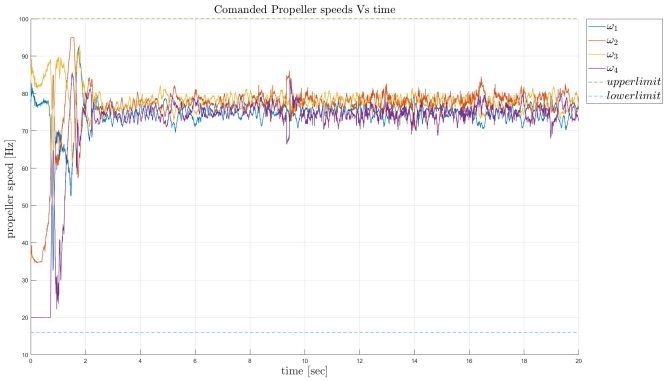
On the other hand, the anticipated outcome of the 'Flying with Loop' experiments was that the quadrotor would face some difficulty finding equilibrium and following the desired trajectory with the given controller gains. The adapted gains were then meant to provide a starting point for designing a gain-scheduling controller that was better adapted to handle the events in the first simulation based on prior information available. Here, however, the quadrotor easily tracks the desired trajectory after an initial period ($\approx$2 s) where the quadrotor settles into its new configuration. Majorly, no additional gain tuning was required to adapt the controller to this new configuration. Additionally, it was observed that the loop was not properly affixed to the hook like in the first scenario, therefore some relative motion was observed between the hook and loop.

The behaviour observed in these two new sets of experiments provides several critical insights. By comparing it against the key characteristics identified it is seen that:

- Changes in Velocity at impact have a significant influence on the quadrotor's behaviour in the 'Impact Isolation' experiments. Higher incidence of crashes occur at higher velocities at impact as observed during the first set of experiments. Changing the velocity at impact for the 'Flying with Loop' Experiments, meanwhile, has no effect even on the settling period observed. The velocity was increased up to 5 m/s in this case to observe if the previously set velocity of 2.5 m/s was too low to observe these effects. The fact that the loop was not properly attached to the hook during the 'Flying with Loop' experiments seems to provide additional evidence that the controller gains are properly tuned.
- A similar forward tilt is observed in the 'Impact Isolation' experiments as seen in the Loop and H simulation scenarios. The magnitude of tilt similarly depends on the velocity at impact.



(a) 'Impact Isolation' Experiment



(b) 'Flying with Loop' Experiment

Figure C.4: Commanded Propeller Speeds

- The commanded propeller speed graphs (Figure C.4) from the 'Impact Isolation' experiments look very similar to the ones from the first set of experiments. Although the commanded propeller speeds seem to touch saturation levels during the initial phase in the 'Flying with Loop' experiments, they soon recover towards a stable equilibrium for the rest of the simulation run.

- The quadrotor with loop configuration settles at a slightly higher set of propeller speeds than the one with just the quadrotor. However, both sets of speeds are well within the range of the propeller speeds defined by the controller. It is interesting to note that no distinguishing behaviour was observed in the new set of simulations as reported in the earlier set (see Section C.1).

The above observations point to the inference that the impact sub-event is at the very least the dominant cause of the post-impact behavior seen during the first set of experiments. The initial placement of the loop during the 'Flying with Loop' experiments is not very accurate. As a result, the loop moves around for the entire duration of the simulation. This suggests that the relative motion of the loop with respect to the hook is of little consequence and the controller is resilient to its effects.

C.5 Counterarguments and Other Key Observations

This section explores the possible counterarguments against the inferences drawn in the previous section. These arguments arise from the fact that the sub-events identified in Section C.1 cannot be truly isolated and replicated in simulation scenarios. This is especially true about the 'Carrying' sub-event. One of the main factors identified in this sub-event was the acceleration (and subsequent deceleration) of the loop (object) around the time of impact. This is a direct consequence of the impact and specific to that particular scenario. It is almost impossible to replicate its occurrence without the impact. Thus the 'Flying with Loop' experiments only replicate part of the factors affecting the Carrying sub-event.

The experiments with H (with clamp: object modified such that impact occurs close to its CoM) clearly show that the objects' physical properties are of some consequence. However, it is interesting to note that the relative motion between the loop and hook has little effect on the quadrotor's ability to track the desired trajectory in the 'Flying with Loop' experiments. This may mean that the physical properties of the object have a small influence on the controller. The additional torque generated about the CoM of the object plays a role in exacerbating the effects of the perturbation caused due to the impact. This may be why using an H-shaped object reduces the instances of crashes.

The commanded propeller speed graphs obtained from the 'Flying with Loop' experiments show that the quadrotor with loop configuration takes a little longer ($\approx$2 s) to settle than just the quadrotor ($\approx$0.6 s). The additional time can be seen as the time taken by the controller to adjust to the payload it has to carry. This will definitely play a role in the Loop and H simulation scenarios as well. In the initial few seconds in Figure C.4b it can be seen that the propeller speeds of both ω_2 and ω_4 decrease. Similar behaviour is observed in Figure 3.4b at the time of impact from the first set of experiments. However, in the 'Impact Isolation' event ω_2 and ω_4 are seen to compensate each other (Figure C.4a). This may suggest that this feature is a characteristic unique to carrying the loop.

The 'Impact Isolation' experiments were specifically designed to be single-point impacts. Therefore this calls into question the impact type in the first set of experiments. Although the scenario is designed for a single-point impact, through the use of a sharp hook, it is not clear what point of the hook impacts the loop and whether there are any subsequent impacts afterwards. Although a small change, it is also important to note that the hook was modified for the 'Impact Isolation' experiments.

C.6 Inferences

A number of interesting insights arose from the above discussion. A lot of these points are beyond the scope of this work. From the perspective of this work, the evidence suggests that the impact sub-event is indeed the most dominant cause of the post-impact behaviour seen in the first simulation scenario. The second most dominant cause is likely the acceleration of the object. However, there is no easy way of isolating this cause to study it further. Experiments with the object modified such that impact occurs close to its CoM provide some evidence of this being the case. The other factors seem to play a much smaller role than anticipated.

Appendix D

Investigating the Noise

Often in this work, multiple simulation trials are conducted with the same conditions. It was observed that the quadrotor's behaviour changed between trials. This change in behaviour was more significant when running the simulation for velocities at impact falling under Class 2 as per classifications detailed in Appendix C. The post-impact behaviour in this Class is significantly affected by the noise. Although, this Class is not the focus of this project. It was essential to investigate the noise since it could affect the integrity of all the simulation tests.

D.1 Approach and Observations

The first step taken in this investigation was to identify possible sources from which the noise could originate. The possible sources were narrowed down to:

- The initial position of the quadrotor
- The sensors conveying information from the Gazebo environment to Simulink
- The trajectory generator

Next, each of these sources was isolated and tested through a series of simulation scenarios as explained below:

D.1.1 Noise from position of quadrotor

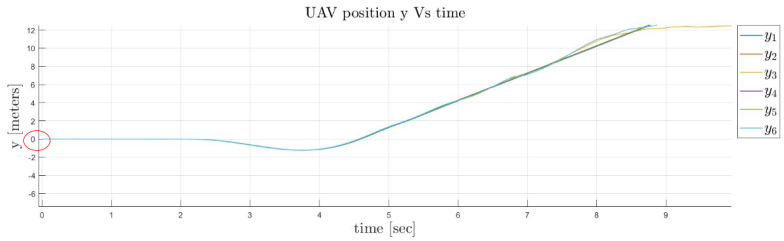
The noise in the position of the quadrotor was observed to originate from two different sources, namely:

- The addition of the passive manipulator below the body of the quadrotor left it without a stable base before it started flying.
- The states of the quadrotor are tracked by the controller through measurements from Optitrack and IMU sensors. Although virtual, the sensors are designed to provide realistic readings and therefore include some measurement noise.

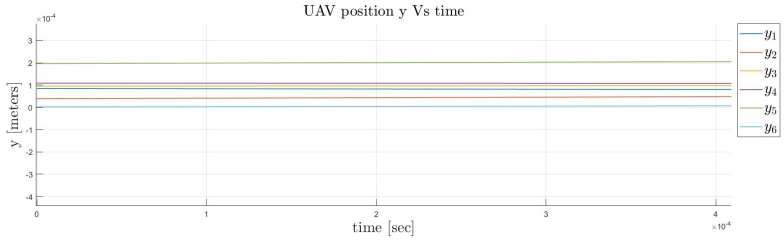
The first cause was easy to observe since the quadrotor always took some time after spawning in the Gazebo environment before it stabilized on the base of the passive manipulator. This implied that the initial position of the quadrotor would change slightly across different trials.

This noise is the easiest to observe and correct. To obtain uniform results across trials, the quadrotor is spawned atop a hollow cylinder that acts as a launch pad as shown in Figure B.2. The quadrotor now rests on its flat body and is unaffected by the design of the passive manipulator.

In order to isolate the second cause, the quadrotor was simulated to run on a fixed uniform linear motion trajectory. The desired trajectory is similar to the trajectory undertaken in all the tests in this work without the impact event. This simulation scenario was run six times and the results were recorded.



(a)



(b)

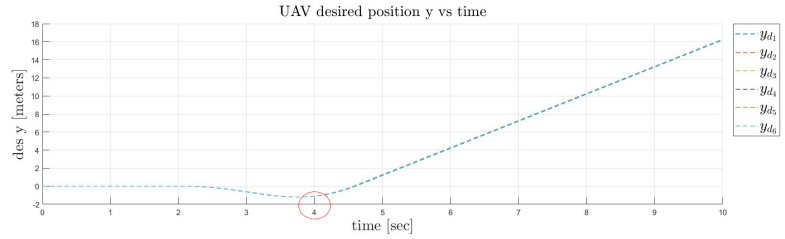
Figure D.1: (a). Quadrotor position along the y-axis across six trials for the same simulation input conditions. The y-axis position is representative of the quadrotor's initial position. (b). Zoomed in version of (a) highlighting the difference in initial position.

Figure D.1 shows the results from the six trials. These trials are conducted using the launch pad in order to eliminate the difference in initial positions caused by the quadrotor balancing atop the passive manipulator. Despite this, there is a difference in the initial position in the order of 10^{-4} m. The source of this noise has to be the measurement noise from the Optitrack and IMU.

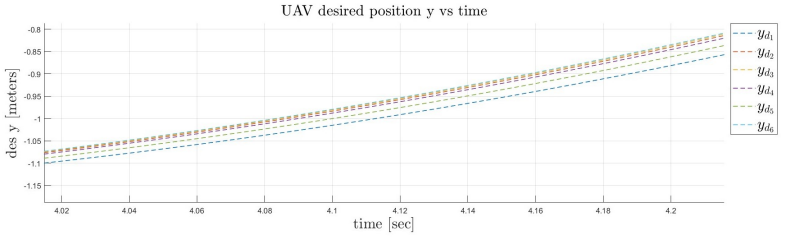
D.1.2 Noise from the desired trajectory

Figure D.2 shows the desired trajectory across the six trials from the previous test. It can be seen that the desired trajectory changes slightly across trials for the same input conditions. The desired uniform linear motion trajectory selected for all simulation scenarios in this work consists of two parts. Initially, the quadrotor achieves the desired velocity using the Simulink Polynomial Trajectory Generator and then the uniform linear trajectory is achieved through a separate code block. This code block takes in the last known position of the quadrotor where it achieves the desired velocity and executes uniform linear motion at the desired velocity. Any possible noise therefore originates in the Simulink Polynomial Trajectory Generator. In order to isolate this noise, the desired trajectory sketched out by the Simulink Polynomial Trajectory Generator is recorded for the same set of inputs. The results from three trials were recorded.

Figure D.3 shows the output of the Simulink Polynomial Trajectory Generator Block for three different trials with the same input parameters. The chosen desired trajectory only varies along the y and z axes. The variations in the output are highlighted. It is clear that the Simulink block generates slightly different polynomial solutions between two input waypoints.



(a)



(b)

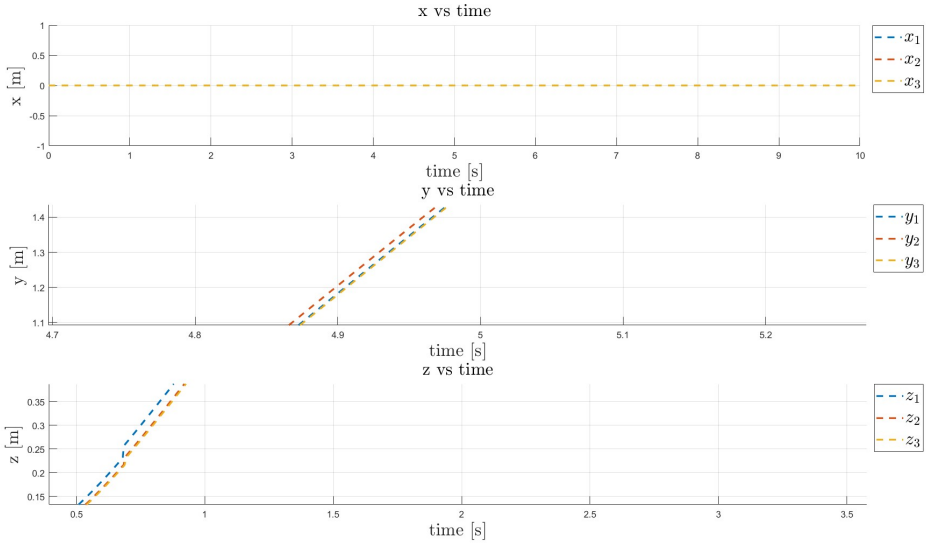
Figure D.2: (a). Desired quadrotor position along the y-axis across six trials for the same simulation input conditions. The y-axis desired position is representative of the changes observed across the desired trajectory. (b). Zoomed in version of (a) highlighting the difference in the desired trajectory.

D.2 Discussion

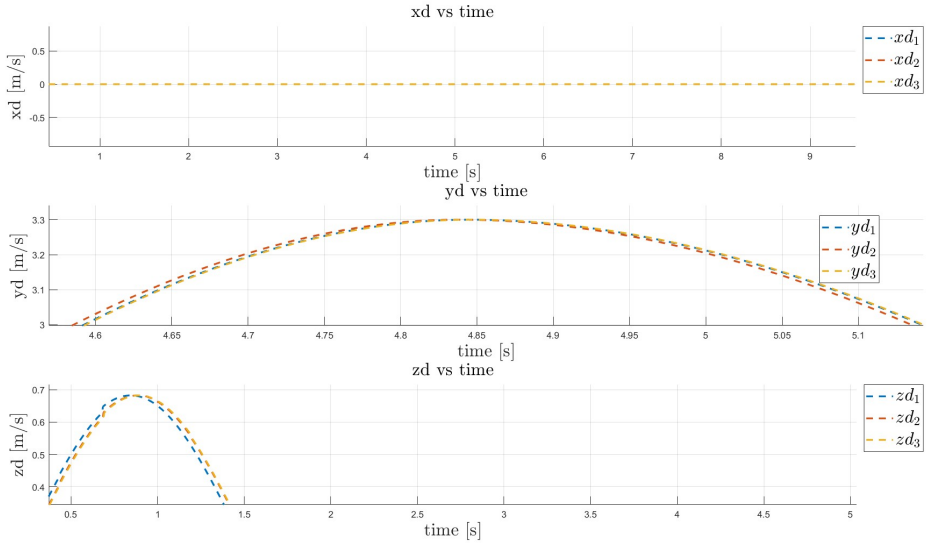
Of the three sources identified two sources can be easily eliminated. Spawning the quadrotor on a launch pad in the Gazebo environment ensures that the quadrotor starts at the same initial position for each trial and therefore effectively eliminates any differences in behaviour that arise from that source. There are two possible methods of eliminating the difference in the output of the Simulink Polynomial Trajectory Generator. One possible method is to provide waypoints that are close to each other as input to the block. The other solution is to save the output from one trial and use that output in all subsequent trials. The second method guarantees to eliminate the noise generated in behaviour as a result of slightly different desired trajectories. However, it was decided that the resultant noise from the desired trajectory sensors helped make the simulations run more realistic. Therefore no action was taken to eliminate the noise emanating from these two sources.

D.3 Conclusions

This Appendix discusses the investigation into the noise observed which leads to a difference in behaviour between simulation trials with the same input parameters. The main sources of noise were identified as the unstable manipulator base, the noise in sensor measurements and the different solutions generated by the Polynomial Trajectory block for the same input. A launchpad was designed to eliminate any changes in the initial quadrotor position resulting from resting on the unstable manipulator base. The other two sources were untouched so as to make the simulations more close to reality.



(a) Desired Position



(b) Desired Velocity

Figure D.3: Output of the Simulink Polynomial Trajectory Generator across three trials for the same input. The variations in the output along the y and z axes are zoomed into and highlighted. There is no variation in the desired trajectory along the x-axis because the trajectory is defined along the y-and z-axes.

Bibliography

[1] M. T. Mason and K. M. Lynch, "Dynamic manipulation," in *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 152–159, 7 1993.

[2] J. J. van Steen, N. van de Wouw, and A. Saccon, "Robot control for simultaneous impact tasks via quadratic programming-based reference spreading," in *2022 American Control Conference (ACC)*, pp. 3865–3872, 2022.

[3] S. S. M. Salehian and A. Billard, "A dynamical-system-based approach for controlling robotic manipulators during noncontact/contact transitions," *IEEE Robotics and Automation Letters*, vol. 3, pp. 2738–2745, 10 2018.

[4] T.-J. Tarn, Y. Wu, N. Xi, and A. Isidori, "Force regulation and contact transition control," *IEEE Control Systems Magazine*, vol. 16, no. 1, pp. 32–40, 1996.

[5] D. Heck, A. Saccon, N. van de Wouw, and H. Nijmeijer, "Switched position-force tracking control of a manipulator interacting with a stiff environment," in *2015 American Control Conference (ACC)*, pp. 4832–4837, 2015.

[6] L. Roveda, N. Iannacci, F. Vicentini, N. Pedrocchi, F. Braghin, and L. M. Tosatti, "Optimal impedance force-tracking control design with impact formulation for interaction tasks," *IEEE Robotics and Automation Letters*, vol. 1, pp. 130–136, 1 2016.

[7] P. R. Pagilla and B. Yu, "A stable transition controller for constrained robots," *IEEE/ASME Transanctions on Mechatronics*, vol. 6, pp. 65–74, 3 2001.

[8] P. R. Pagilla and M. Tomizuka, "Contact transition control of nonlinear mechanical systems subject to a unilateral constraint," *Journal of Dynamic Systems, Measurement, and Control*, vol. 119, pp. 749–759, 1997.

[9] M. Martino and M. E. Broucke, "A reach control approach to bumpless transfer of robotic manipulators," in *53rd IEEE Conference on Decision and Control*, pp. 25–30, 2014.

[10] F. Ferraguti, C. Secchi, and C. Fantuzzi, "A tank-based approach to impedance control with variable stiffness," in *2013 IEEE International Conference on Robotics and Automation*, pp. 4948–4953, 2013.

[11] M. Rijnen, A. Saccon, and H. Nijmeijer, "Reference spreading: Tracking performance for impact trajectories of a 1dof setup," *IEEE Transactions on Control Systems Technology*, vol. 28, pp. 1124–1131, 5 2020.

[12] J. J. van Steen, A. Coşgun, N. van de Wouw, and A. Saccon, "Dual arm impact-aware grasping through time-invariant reference spreading control," 12 2022.

[13] J. J. van Steen, N. van de Wouw, and A. Saccon, "Robot control for simultaneous impact tasks through time-invariant reference spreading," 6 2022.

[14] W. Yang and M. Posa, "Impact-invariant control: Maximizing control authority during impacts," 3 2023.

[15] I. Aouaj, V. Padois, and A. Saccon, "Predicting the post-impact velocity of a robotic arm via rigid multibody models: An experimental study," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 12639–12645, Institute of Electrical and Electronics Engineers Inc., 2021.

[16] W. Wouter, "Validation of nonsmooth impact maps in robot impact experiments," Master's thesis, Eindhoven University of Technology, 4 2021.

[17] B. Proper, A. Kurdas, S. Abdolshah, S. Haddadin, and A. Saccon, "Aim-aware collision monitoring: Discriminating between expected and unexpected post-impact behaviors," Master's thesis, TU/e, 2021.

[18] M. W. Rijnen, A. T. van Rijn, H. Dallali, A. Saccon, and H. Nijmeijer, "Hybrid trajectory tracking for a hopping robotic leg," in *IFAC-Papers*, vol. 49, pp. 107–112, Elsevier B.V., 2016.

[19] M. Rijnen, E. D. Mooij, S. Traversaro, F. Nori, N. V. D. Wouw, A. Saccon, and H. Nijmeijer, "Control of humanoid robot motions with impacts: Numerical experiments with reference spreading control," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4102–4107, Institute of Electrical and Electronics Engineers Inc., 7 2017.

[20] S. J. Uitendaal, "Msc thesis teaching robots impact tasks by performing demonstrations," Master's thesis, TU Delft, 2022.

[21] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se(3)," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 5420–5425, 2010.

[22] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2247–2252, 2005.

[23] N. Guenard, T. Hamel, and V. Moreau, "Dynamic modeling and intuitive control strategy for an "x4-flyer"," in *2005 International Conference on Control and Automation*, vol. 1, pp. 141–146 Vol. 1, 2005.

[24] F. Bullo and A. D. Lewis, *Geometric Control of Mechanical Systems*, vol. 49 of *Texts in Applied Mathematics*. New York-Heidelberg-Berlin: Springer Verlag, 2004.

[25] M. Ryll, G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, D. Bicego, and A. Franchi, "6D interaction control with aerial robots: The flying end-effector paradigm," *The International Journal of Robotics Research*, vol. 38, pp. 1045–1062, Aug. 2019. Publisher: SAGE Publications Ltd STM.

[26] T. Tomić, C. Ott, and S. Haddadin, "External wrench estimation, collision detection, and reflex reaction for flying robots," *IEEE Transactions on Robotics*, vol. 33, pp. 1467–1482, 12 2017.

[27] F. Ruggiero, V. Lippiello, and A. Ollero, "Aerial Manipulation: A Literature Review," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1957–1964, July 2018. Conference Name: IEEE Robotics and Automation Letters.

[28] V. Ghadiok, J. Goldin, and W. Ren, "Autonomous indoor aerial gripping using a quadrotor," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4645–4651, Sept. 2011. ISSN: 2153-0866.

[29] M. Fumagalli, R. Naldi, A. Macchelli, R. Carloni, S. Stramigioli, and L. Marconi, "Modeling and control of a flying robot for contact inspection," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3532–3537, Oct. 2012. ISSN: 2153-0866.

[30] T. Bartelds, A. Capra, S. Hamaza, S. Stramigioli, and M. Fumagalli, "Compliant Aerial Manipulators: Toward a New Generation of Aerial Robotic Workers," in *IEEE Robotics and Automation Letters*, vol. 1, pp. 477–483, Jan. 2016.

[31] H. Bonyan Khamseh, F. Janabi-Sharifi, and A. Abdessameud, "Aerial manipulation—A literature survey," *Robotics and Autonomous Systems*, vol. 107, pp. 221–235, Sept. 2018.

[32] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, Present, and Future of Aerial Robotic Manipulators," *IEEE Transactions on Robotics*, vol. 38, pp. 626–645, Feb. 2022.

[33] C. Papachristos, K. Alexis, and A. Tzes, "Efficient force exertion for aerial robotic manipulation: Exploiting the thrust-vectoring authority of a tri-tiltrotor UAV," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4500–4505, May 2014. ISSN: 1050-4729.

[34] K. Bodie, M. Brunner, M. Pantic, S. Walser, P. Pfändler, U. Angst, R. Siegwart, and J. Nieto, "Active Interaction Force Control for Contact-Based Inspection With a Fully Actuated Aerial Vehicle," *IEEE Transactions on Robotics*, vol. 37, pp. 709–722, June 2021. Conference Name: IEEE Transactions on Robotics.

[35] L. A. Tony, S. Jana, V. V. P., A. A. Bhise, A. M. V. S., V. B. V., M. S. Gadde, R. Krishnapuram, and D. Ghose, "Autonomous Cooperative Multi-Vehicle System for Interception of Aerial and Stationary Targets in Unknown Environments," Sept. 2021. arXiv:2109.00481 [cs, eess].

[36] B. V. Vidyadhara, L. A. Tony, M. S. Gadde, S. Jana, V. P. Varun, A. A. Bhise, S. Sundaram, and D. Ghose, "Design and integration of a drone based passive manipulator for capturing flying targets," *Robotica*, vol. 40, pp. 2349–2364, July 2022. Publisher: Cambridge University Press.

[37] J. Thomas, J. Polin, K. Sreenath, and V. Kumar, "Avian-Inspired Grasping for Quadrotor Micro UAVs," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. Volume 6A: 37th Mechanisms and Robotics Conference, p. V06AT07A014, 08 2013.

[38] J.-P. Ore, S. Elbaum, A. Burgin, and C. Detweiler, "Autonomous Aerial Water Sampling," *Journal of Field Robotics*, vol. 32, no. 8, pp. 1095–1113, 2015. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21591.

[39] B. Siciliano and O. Khatib, eds., *Springer Handbook of Robotics*. Springer Handbooks, Cham: Springer International Publishing, 2016.

[40] S. Parusel, S. Haddadin, and A. Albu-Schäffer, "Modular state-based behavior control for safe human-robot interaction: A lightweight control architecture for a lightweight robot," in *2011 IEEE International Conference on Robotics and Automation*, pp. 4298–4305, 2011.

[41] T. Tomić and S. Haddadin, "Simultaneous estimation of aerodynamic and contact forces in flying robots: Applications to metric wind estimation and collision detection," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5290–5296, 2015.

[42] Y. Gong and J. Grizzle, "Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller," *arXiv preprint arXiv:2008.10763*, 2020.

[43] M. Hamandi, F. Usai, Q. Sable, N. Staub, M. Tognon, and A. Franchi, "Design of multirotor aerial vehicles: a taxonomy based on input allocation," *The International Journal of Robotics Research*, 7 2021.

Abbreviations

AAU Atomic Actuation Unit. 41

CCW Counter Clockwise. 41

CoM Center of Mass. 5, 6, 9, 10, 14, 21, 22, 30, 35, 38, 40, 43, 51, 52

CW Clockwise. 41

DoF degrees of freedom. 5, 6, 21, 40

GenoM Generator of Modules. 11

MAV Multirotor Aerial Vehicles. 5, 41

RSHC Reference Spreading Hybrid Control. 2

UAV Unmanned Aerial Vehicle. 5–7, 12, 13, 16, 18, 38, 42