Stochastic Optimal Shared Control with Non-Gaussian Processes

Paul Onor
UNM

Follow this and additional works at: https://digitalrepository.unm.edu/ece_etds

Part of the Electrical and Computer Engineering Commons

Recommended Citation
https://digitalrepository.unm.edu/ece_etds/536

This Thesis is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Electrical and Computer Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.
Paul Onor

Candidate

Electrical Engineering

Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Dr Meeko Oishi

Chair

Dr Rafael Fierro

Member

Dr Claus Danielson

Member
Stochastic Optimal Shared Control with Non-Gaussian Processes

BY

Paul Onor
B.S., Electrical Engineering, University of New Mexico, 2020

THESIS

Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Science

Electrical Engineering
The University of New Mexico
Albuquerque, New Mexico

May 2022
DEDICATION

I dedicate this thesis to my mother, Lisa, who has supported me every step of the way and to my father, Zaf, who is no longer with us. I know you’re proud, wherever you are, and I really wish you could be here still.
I want to express my gratitude to everyone who has helped me along the journey to this point.

Especially, I want to thank my advisor Dr Oishi for having patience with me during this research process and keeping me on track to completion, as well as Vignesh from the HCPS lab for assisting me with the background research and theory.

My gratitude goes out to the many professors who gave me the skills and knowledge to be here in the first place. Whether directly applicable skills or baseline knowledge, I wouldn’t have been able to do it without you.

Finally, I want to thank my loving mother as well as my wonderful friends for their support and reassurance.

This material is based upon work supported by the National Science Foundation under NSF Grant Number CNS-1836900. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.
Stochastic Optimal Shared Control with Non-Gaussian Processes

by

Paul Onor

B.S., Electrical Engineering, University of New Mexico, 2020

M.S., Electrical Engineering, University of New Mexico, 2022

In this paper, we propose an approach for shared control of a planar quadrotor that allows for non-Gaussian disturbance in the model and non-Gaussian variation in the pilot’s control actions. We do this by constructing empirical characteristic functions for the state, inputs, and disturbance using demonstrations by a human expert. These are then used to make predictions of future states and of the system disturbance, using the first and second moments of the empirical characteristic function to estimate the mean and variance of these processes. With this method, we can extend assumptions from additive white Gaussian noise to any real-valued disturbance for a system with a quadratic cost. The proposed method is shown to properly find a control scheme and maintain system performance with a generalized disturbance model.
Contents

List of Figures viii

List of Tables x

1 Introduction 1

1.1 Motivation and Background 2

1.2 Problem Statement 4

1.3 Statement of Contribution 4

1.4 Outline 5

2 Mathematical Preliminaries and Problem Setup 6

2.1 Notation 6

2.2 System Overview 6

2.3 Plant Dynamics 7

2.4 Problem Formulation 8

2.5 Characteristic Functions 9

2.6 Kalman Filtering 12

2.6.1 Standard Kalman Filter 13

2.6.2 Kalman Filter using Characteristic Functions 14

2.6.3 Non-Gaussian (Generic) Filter 15
# 3 Methods

3.1 Inferring Expert Policy ........................................... 17
3.2 Shared Control ..................................................... 21

# 4 Results and Discussion

4.1 Experimental Setup .................................................. 23
4.2 Initial Data ........................................................... 24
   4.2.1 Human Expert .................................................... 24
   4.2.2 Simulated Novices .............................................. 25
4.3 Simulation Setup ..................................................... 25
   4.3.1 Simulated Plant Dynamics ...................................... 25
   4.3.2 Simulated Disturbance Models ................................. 26
4.4 Simulation Results ................................................... 27
4.5 Future Work .......................................................... 31
   4.5.1 Limitations ........................................................ 31
   4.5.2 Expansion to 3D ................................................. 31

# 5 Conclusion

# A Plant Model Derivation

A.1 Nonlinear Model .................................................... 37
A.2 Linearized Model ................................................... 38
A.3 Disturbance .......................................................... 39
A.4 Final State Space Model ............................................ 41
## List of Figures

2.1 Shared control system overview from [1] ............................................. 7

2.2 Convergence of estimated statistics to true statistics for a standard normal distribution ............................................................ 11

2.3 Pdf and cdf evolution with increasing number of samples, roughly 30 samples corresponds to 1 second of data collection in our system .... 12
   a 5 samples ................................................................. 12
   b 15 samples ............................................................. 12
   c 50 samples .............................................................. 12
   d 100 samples ............................................................ 12

4.1 2D landing simulator for data collection and human subject testing .... 23

4.2 Human demonstrations with mean demonstration used for IRL ....... 24

4.3 Noise profiles used for simulation ................................................. 27
   a Uniform distributions used: U(-0.1, 0.1), U(-1, 1), U(0, 1), U(0.5, 1.5) ......................................................... 27
   b Beta distributions used: $\beta(5, 1), \beta(1, 5), \beta(2, 3)$ ...................... 27
   c Gamma distributions used: $\Gamma(1, 3), \Gamma(3, 1)$ ......................... 27
   d Laplace distributions used: Laplace(0, 1), Laplace(0, 2) ................. 27

4.4 Landing Distance to Origin and Control Authority Statistics .......... 29
   a Simulation results for distance to the origin, averaged with respect to the 20 novices ......................................................... 29
b Control authority statistics averaged across novices . . . . . . 29

4.5 Mean trajectory of each noise profile for the expert, unaided novice,
and novice aided by shared control . . . . . . . . . . . . . . . . . . . . . 30
# List of Tables

4.1 Cumulative Cost Statistics ........................................... 30

A.1 Quadrotor constants .................................................. 41
Chapter 1

Introduction

Cyberphysical systems are becoming ever-present in modern life. From manufacturing to airplanes to smart grids, we interact with them on a daily basis; yet, the interactions between automation and humans is hard to perfect to ensure smooth and safe operation under uncertain conditions. This is in part due to human actions being complex and heterogeneous, even among users of comparable skill levels. Additionally, determining optimal operation via computation in the presence of noise or large state spaces can be challenging, but training a human to act more-or-less optimally in a variety of situations based on experience is possible in these types of systems (e.g., driving a car or flying a plane). We can apply inverse reinforcement learning techniques to capitalize upon this and take the best aspects of a human operator to teach our automated system to handle uncertain situations.

The UNM cyberphysical systems lab and our collaborators at Purdue are interested in refining the methods for sharing control between human operators and automated systems to ensure safer and more stable operation. We would like to do so in a manner that leaves human operators with as much agency as possible and not impede their control of the system unless it is absolutely necessary to meet the end goal.
1.1 Motivation and Background

As automation continues to expand into the public sphere, we must examine the role of human operators and how they interact with the various systems present in daily life. We must examine when an intervention by automated systems is required or when the human operator may be allowed to operate a system suboptimally. To that end, we must find a way to model those actors who may take unexpected actions and actions that vary greatly between individuals. These variations must be accounted for in a system to ensure that it works generally, and that it is not only functional and stable in the hands of an expert, but also in the hands of a novice; ideally, it will help train the novice by showing when control authority is lost and when the novice’s proposed control actions are close to optimal.

While many natural processes do contain Gaussian disturbances, it should not necessarily be the default assumption for the distribution of human actions. We assume that the spectrum of human actions of comparable skill is non-Gaussian and recreate the experiment from [1] with our new assumption; thus, we make it more suitable for general application with unknown process noise and human variability. Our system model for demonstration, a 3 degree of freedom (3Dof) quadrotor, comes from [2], which performs analysis and control of a planar quadrotor in the absence of noise. However, since we intend to add noise, we borrow from the model in [3] which introduces a matrix to describe how a noise vector affects the state vector. The addition of noise also means that the control methods from [1] must be altered slightly, which we explore through the use of the empirical characteristic function (ECF) [4], [5]. We could alternately make assumptions on our non-Gaussian distribution and construct a Kalman-type filter [6], [7] for estimating the noise mean and covariance and allowing control, but in the interest of keeping our solution as general as possible, we opted for the ECF approach that makes no assumptions about the distribution. Our chosen method for computing the necessary process moments from the ECF,
evaluation at zero of the first and second derivatives, contrasts with [8] which presents a method for estimating the probability distribution of a series of data by numerical inversion.

Shared control for this setup is well explored in [9], [10], and [11]. We follow the methodology from [1] but exchange the offline portion of the calculations and the inverse optimal control with our robust formulation. Notably, we do not use the online dynamic mode decomposition (DMD) from [12] that is applied in [1] to compute the inverse reinforcement learning, and instead we follow methods from [10], [13], [14], [15], [16], and [17] to learn the human demonstrator’s approximate weight matrices and infer new control inputs from new states. Because we are sharing control based on discrete sharing ratios, we additionally consult [18], [19], and [20] for exploring methods of inverse optimal control with modal cost functions, e.g., when the automation system intervenes, and under stochasticity in our system, the cost function swaps between discrete modes. Typically, we can derive a linear-quadratic-Gaussian (LQG) controller using the algebraic Riccati equation (ARE) by iterative methods to determine feedback gain, but this is much more difficult when the disturbance is unknown. When there is Gaussian uncertainty in the system, we can use Gaussian mixture models to synthesize a new control input [21] from the current observed state, or using a linear quadratic regulator with Gaussian noise [22], but we synthesize our inputs using optimal control with a penalty on the automation intervention, following the framework laid out in [1]. Another solution for stochastic model predictive control is proposed in [23] where they discuss a similar method to the one presented without the use of the ECF. Our solution differs because we are finding an optimal solution with arbitrary system disturbance while also sharing control with a human operator.

In contrast to the existing literature, we propose a framework for performing inverse reinforcement learning using the empirical characteristic function to compute process mean and covariance of an arbitrary system disturbance. We demonstrate
the effectiveness of this approach and show that it does in fact improve system performance in the face of unknown disturbance.

### 1.2 Problem Statement

We would like to answer the main question,

MQ1: How can we extract the expert’s control policy from a sequence of observations without knowing the system disturbance?

to which we will add one sub-question,

SQ2: Can we find the expectation of the quadratic cost when the disturbance distribution is unknown?

First, we must discuss how to derive an optimal control policy without knowing the disturbance. We use the empirical characteristic function, which takes a sequence of samples and estimates the Fourier transform of the probability density function of the underlying distribution. Then we can find central moments to calculate necessary statistics such as the mean and variance for the unknown distribution.

Second, we discuss how to perform inverse optimal control in the face of disturbances and use this to infer our control policy for shared control. We would like to use a quadratic cost for our system, but with disturbance we are unable to predict the true cost at any future time step. Instead, we must predict the expectation of the cost and use this to infer the optimal control policy instead.

We will formalize these in Section 2.4.

### 1.3 Statement of Contribution

The main novelty of this work is:
1. Extending an existing framework for optimal shared control with Gaussian noise to account for non-Gaussian processes for a quadratic cost

2. Performing inverse optimal control using the empirical characteristic function to estimate noise statistics

My contribution to this work was to a) derive the inverse quadratic cost with assistance from a senior graduate student, b) implement our proposed algorithms in Matlab, and c) evaluate the method numerically using extensive simulation.

This contribution is in preparation to be submitted to the American Control Conference 2023: P. Onor, V. Sivaramakrishnan, S. Byeon, M. Oishi, I. Hwang, “Stochastic Optimal Shared Control with Non-Gaussian Processes,” ACC 2023, to be submitted

1.4 Outline

The layout of the remainder of this thesis is as follows: Chapter 2 discusses some mathematical preliminaries as well as the problem formulation, Chapter 3 discusses our proposed methods for stochastic inverse optimal control and shared control, Chapter 4 demonstrates the proposed policy and methods in a simulated example and discusses future improvements, and Chapter 5 provides some concluding remarks.
Chapter 2

Mathematical Preliminaries and Problem Setup

2.1 Notation

In this paper, we will use the following notation. We indicate a random vector with a bold, lowercase $\mathbf{w} \in \mathbb{R}^n$ and use an uppercase $A \in \mathbb{R}^{n \times m}$ for a matrix. We denote a sequence as $\mathbb{N}_{[a,b]}$, $a, b \in \mathbb{N}$, $b > a$. A positive definite and positive semi-definite matrix are denoted $Q \succ 0$ and, $R \succeq 0$ respectively. For the imaginary unit, we use $j = \sqrt{-1}$ throughout the paper.

2.2 System Overview

We presume a system model consisting of an offline estimation of an expert control policy and objective function, an online estimate of the current user’s control policy, a plant to be controlled with some unknown additive disturbance, and a controller to decide how control authority is being shared during various modes of operation (Fig.2.1, [1]). The hybrid shared controller is described in [1], with trivial changes being made in this work; thus, we will focus our energy on the reformulation of the
inverse optimal control (IOC) blocks for the remainder of this work.

2.3 Plant Dynamics

We will be examining a discrete-time linear time-invariant (LTI) system of the form:

\[
x[k + 1] = Ax[k] + Bu[k] + Dd[k]
\]  

(2.1)

with state vector \( x \in \mathbb{R}^n \), control input \( u \in \mathbb{R}^m \), and disturbance \( d \in \mathbb{R}^p \) with matrices \( A \), \( B \), and \( D \) having appropriate dimensions.

Additionally, we assume the state-space system is controllable and that we have perfect state information, i.e., the state observation matrix is an identity matrix, so that the problem is well posed.
2.4 Problem Formulation

We must estimate the expert’s control policy from a sequence of observations of the state and input, which we denote \((x_e, u_e)_{[1,N]}\), then we must use this estimate in real-time to share authority with a novice using the system. The authority sharing problem has been solved already in [1], but we formalize below the problem of inferring the expert’s control policy in the presence of non-Gaussian disturbances.

**Problem 1** (Estimate expert control policy). *Presuming that a human expert has a linear feedback control policy of the form:*

\[
    u_{e,k} = K_e x_k
\]

*which gives an optimal cost when applied to a quadratic cost function*

\[
    J_k(x_e, u_e) = x_{e,k+1}^T Q x_{e,k+1} + u_{e,k}^T R u_{e,k}
\]

*subject to the dynamics in (2.1) with \(Q \in \mathbb{R}^{nxn} \succ 0\) and \(R \in \mathbb{R}^{mxm} \succeq 0\). We will estimate the weighting matrices \(Q, R\) using a series of demonstrations by the expert.*

The inverse linear quadratic regulation (LQR) problem without noise can be solved via a set of linear matrix inequalities. More specifically, it results in a semi-definite feasibility problem [9], [14].

\[
\text{find } \{Q_e, R_e\} \\
\text{subject to } Q_e + A^T P_k (A + B K_e) - P_k = 0 \hspace{1cm} (2.4a) \\
R_e + B^T P_k (A + K_e) = 0 \hspace{1cm} (2.4b) \\
P_k \succeq 0, \ Q_e \succeq 0, \ R_e \succeq 0 \hspace{1cm} (2.4d)
\]

which we can solve using standard methods. However, when we introduce uncertainty,
we have to reformulate the problem to account for the state becoming a random variable. We also cannot use Kalman filtering, [6], [7], because we are not assuming a specific model for our noise; instead, we will use trajectories of the system to estimate statistics of the disturbance without injecting any knowledge about its form. This is discussed in more detail in the following chapters.

We will then use the results of problem 1 to apply the method of shared control presented in [1] to demonstrate this controller in Monte-Carlo simulations.

### 2.5 Characteristic Functions

**Definition 1** (Characteristic Function [4]). *Given a random vector \( \mathbf{w} \in \mathbb{R}^p \) with known probability density function \( \Phi_{\mathbf{w}}(x) \), the characteristic function of \( \mathbf{w} \) is*

\[
\phi_{\mathbf{w}}(t) = \mathbb{E}[e^{jt\mathbf{w}}] = \int_{\mathbb{R}^p} e^{jt\mathbf{w}} d\Phi_{\mathbf{w}}(x) 
\]

which is the Fourier transform of \( \Phi_{\mathbf{w}}(x) \) with \( t \in \mathbb{R} \) as the transform variable.

With an unknown underlying distribution, or in the absence of a closed-form expression for the pdf of the underlying distribution, we instead must find the empirical characteristic function which is a weighted sum of exponentials of the data combined with a smoothing matrix.

**Definition 2** (Empirical Characteristic Function [24], [25]). *Given a sequence of \( N \) observations, \( \mathbf{w}_{[1,N]} \), of a random vector \( \mathbf{w} \), the empirical characteristic function of \( \mathbf{w} \) is*

\[
\hat{\phi}_{\mathbf{w}}(t) = \sum_{k=1}^{N} \sigma_k(\mathbf{w}) K_{\mathbf{w}_k}(t) 
\]

\[
K_{\mathbf{w}_k}(t) = e^{(jt\mathbf{w}_k)} e^{-\frac{1}{2}t\Sigma_k} 
\]

9
for a weighting function $\sigma$ such that $\sum_{k=1}^{N} \sigma_k = 1$ and suitably chosen smoothing matrix $\Sigma$.

The smoothing matrix must be chosen to ensure continuity of the probability distribution function found by inverting the ECF. There are many methods for choosing a smoothing matrix, or smoothing value in the 1-dimensional case, which are discussed in detail in [26]. Our chosen method is based on solutions to the partial differential equation describing linear diffusion, as described in [27].

With a characteristic function as defined above, each $i^{th}$ moment is given by taking the derivatives of the characteristic function and evaluating at zero.

$$E[w^d] = (-j)^d \left[ \frac{\partial^d}{\partial t_1^d} \varphi_w(t) \ldots \frac{\partial^d}{\partial t_p^d} \varphi_w(t) \right]_{t=0}$$ (2.7)

The only two moments we will need to use are the first moment,

$$E[w] = \left[ \frac{\partial}{\partial t_1} \varphi_w(t) \ldots \frac{\partial}{\partial t_n} \varphi_w(t) \right]_{t=0}. \quad (2.8)$$

and the second moment,

$$E[ww^T] = \left[ \frac{\partial^2}{\partial t_1^2} \varphi_w(t) \ldots \frac{\partial^2}{\partial t_1 \partial t_2} \varphi_w(t) \ldots \frac{\partial^2}{\partial t_1 \partial t_n} \varphi_w(t) \ldots \frac{\partial^2}{\partial t_n^2} \varphi_w(t) \right]_{t=0}. \quad (2.9)$$

as we will demonstrate in the following sections.

Compare this to the standard method of computing moments of a random variable $w$ with pdf $\Phi_w$

$$E[w^d] = \int_{-\infty}^{\infty} w^d \Phi_w dw$$ (2.10)

and we can see that the characteristic function route leads to a computationally easier
formulation as a derivative is easier to compute than an integral in general.

Figure 2.2: Convergence of estimated statistics to true statistics for a standard normal distribution

Additionally, we see in Figure 2.2 that the estimated statistics provided by the ECF converge in only a few hundred samples to the true mean and variance. The cumulative distribution function (cdf) and probability distribution functions (pdf) can be extracted from the characteristic function and also converge to the true distribution functions, though they converge at different rates; the estimated cdf converges rather quickly, around 25 samples, but the estimated pdf takes fewer samples to approximate the true pdf accurately as shown in Figure 2.3. The convergence shown in Figure 2.3 depends greatly on the choice of smoothing matrix. If the value is chosen too large, then the result will obscure smaller features, especially in multi-peak distributions, but chosen too small and the estimated pdf and cdf will appear jagged as they more closely will resemble a collection of Dirac delta functions at the sample data points. The method we chose for selecting a smoothing matrix is designed to change as we add more data, allowing the estimate to adapt more easily and change smoothing dynamically. This can cause the accuracy to vary slightly even with more samples.
2.6 Kalman Filtering

Under standard Gaussian noise, there are methods that can be used to approximate the noise statistics and control the system. The optimal method for this is the Kalman filter, described below. We start with the simplest form of the Kalman filter, as an extension of the optimal linear-quadratic regulator with Gaussian disturbance (LQG) and later show formulations utilizing characteristic functions [5] and a more generic version of the Kalman filter that can approximate specific non-Gaussian disturbances [7].
2.6.1 Standard Kalman Filter

With an LTI system of the form:

\[ x_{k+1} = A_kx_k + B_ku_k + F_kw_k \]  

(2.11)

where the process noise \( w_k \) is zero mean white noise with covariance

\[ E[w_kw_k^T] = Q_k \]  

(2.12)

And measurements:

\[ z_k = H_kx_k + v_k \]  

(2.13)

where \( v_k \) is zero mean white measurement noise with covariance

\[ E[v_kv_k^T] = R_k \]  

(2.14)

First we calculate the measurement residual \( i_k \) as

\[ i_k = z_{k+1} - H\hat{x}_k \]  

(2.15)

Kalman Gain:

\[ K_k = P'_{k-1}HT(HP'_{k-1}HT + R)^{-1} \]  

(2.16)

Update Estimate:

\[ \hat{x}_k = \hat{x}'_k + K_k(z_k - H\hat{x}'_k) \]  

(2.17)

Update Covariance:

\[ P_k = (I - K_kH)P'_k \]  

(2.18)
Project into \( k+1 \):

\[
\hat{x}_{k+1}' = A\hat{x}_k + Bu_k
\]  \hspace{1cm} (2.19)

\[
P_{k+1} = AP_kA^T + Q
\]  \hspace{1cm} (2.20)

These equations are run at each time step and form a progressively narrowing estimate of the state and noise covariance. It is assumed that the noise process has zero mean, so there is no need to estimate it here.

### 2.6.2 Kalman Filter using Characteristic Functions

From [5] we can describe the Kalman filter in terms of characteristic functions. At each step we update the ECF of the state as:

\[
\bar{\phi}_{X_k|Y_k}(\nu) = C_k \exp\left(-\frac{1}{2}\nu^T P_k \nu + j\nu^T \hat{x}_k|k\right)
\]  \hspace{1cm} (2.21)

where

\[
\hat{x}_k|k = \hat{x}_{k|k-1} + K_k(z_k - H\hat{x}_{k|k-1})
\]  \hspace{1cm} (2.22)

\[
P_k = M_k - M_kH^T(hM_kH^T + V)^{-1}HM_k
\]  \hspace{1cm} (2.23)

\[
K_k = M_kH^T(HM_kH^T + V)^{-1}
\]  \hspace{1cm} (2.24)

and

\[
C_k = C_{k-1}f_{R_k}(r_k)
\]  \hspace{1cm} (2.25)

With the process noise characteristic function as:

\[
\phi_W(v_w) = \exp\left(-\frac{1}{2}v_w^T Q v_w\right)
\]  \hspace{1cm} (2.26)
We can propagate forward in time via:

$$
\bar{\phi}_{X_{k+1}|Y_k} = \bar{\phi}_{X_k|Y_k} (\Phi^T v) \phi(F^T v) = C_k \exp \left( -\frac{1}{2} v^T M_{k+1} v + j v^T \hat{x}_{k+1|k} \right) \tag{2.27}
$$

where

$$
\hat{x}_{k+1|k} = A\hat{x}_{k|k} + Bu, \tag{2.28}
$$

$$
M_{k+1} = AP_k A^T + F Q F^T \tag{2.29}
$$

The result is effectively the same as the previous section, but makes use of the characteristic functions of the state and noise process.

### 2.6.3 Non-Gaussian (Generic) Filter

From [5], [7] we can generalize to specific non-Gaussian disturbance types.

Assume the characteristic function of the unnormalized pdf is known:

$$
\bar{\phi}_{X_k|Y_{k-1}}(\nu) = \int_{x_k} \bar{f}_{X_k|Y_{k-1}} \exp j \nu^T x_k \, dx_k \tag{2.30}
$$

then the characteristic function updated with measurement data from time $k$ is:

$$
\bar{\phi}_{X_k|Y_k}(\nu) = \frac{1}{(2\pi)^m} \int_{\xi} \bar{f}_{X_k|Y_{k-1}}(\nu - H^T \xi) \phi_V(-\xi) \exp j \xi^T z_k \, dx_i \tag{2.31}
$$

with

$$
\xi = H^{-T} \eta \rightarrow \eta = H^T \xi \rightarrow d\eta = |H| \, d\xi \tag{2.32}
$$

Propagating in time yields:

$$
\bar{\phi}_{X_{k+1}|Y_k}(\nu) = \bar{\phi}_{X_k|Y_k} (\Phi^T v) \phi_W (\Gamma^T \nu) \tag{2.33}
$$
following

\[ x_{k+1} = \Phi x_k + \Gamma w_k \quad (2.34) \]

Then our conditional mean of state \( x_k \) given data sequence \( y_k \) is

\[
\hat{x}_k = E[x_k|y_k] = \frac{1}{j f_{Y_k}(y_k)} \left( \frac{\partial \tilde{\phi}_{X_k|Y_k}(\nu)}{\partial (\nu)} \right)^T |_{\nu=0} \quad (2.35)
\]

and our second conditional moment is

\[
E[x_k x_k^T|y_k] = \frac{1}{j^2 f_{Y_k}(y_k)} \left( \frac{\partial^2 \tilde{\phi}_{X_k|Y_k}(\nu)}{\partial (\nu) \partial (\nu)^T} \right)^T |_{\nu=0} \quad (2.36)
\]

These are demonstrated to work with Cauchy disturbances in [7] but could in theory be extended to other forms of disturbance (with section 2.6.2 being a special case).
Chapter 3

Methods

3.1 Inferring Expert Policy

Recall Problem 1, in the presence of stochasticity, we cannot optimize over the precise cost due to the uncertainty introduced. Instead, we presume the expert control policy optimizes a quadratic cost of the form:

\[
E[J_k] = E[x_{k+1}^\top Q x_{k+1} + u_k^\top R u_k] \quad (3.1)
\]

subject to (2.2) and the dynamics in (2.1). To solve the inverse problem, we first must be able to estimate the quadratic cost at each stage.

**Theorem 1.** We can find the expected cost \(E[J_k]\) using only the first two moments of the characteristic function.

*Proof:*

First, we expand the expected cost from (3.1) using the trace-trick.

\[
E[J_k] = E[x_{k+1}^\top Q E[x_{k+1}] + u_k^\top R E[u_k] + \text{Tr}(Q \Sigma_{x_{k+1}}) + \text{Tr}(R \Sigma_{u_k}) \quad (3.2)
\]
with $\Sigma_{x_{k+1}}$ and $\Sigma_{u_k}$ defined as follows.

$$
\Sigma_{x_{k+1}} = \mathbb{E}[x_{k+1}'x_{k+1}'] - \mathbb{E}[x_{k+1}']\mathbb{E}[x_{k+1}]
$$

(3.3a)

$$
\Sigma_{u_k} = \mathbb{E}[u_k'u_k'] - \mathbb{E}[u_k']\mathbb{E}[u_k]
$$

(3.3b)

Recall now definition 2 and equations (2.8) and (2.9). Combining these, we can compute $\mathbb{E}[x_{k+1}], \mathbb{E}[x_{k+1}'x_{k+1}'], \mathbb{E}[u_k], \text{ and } \mathbb{E}[u_k'u_k']$. Thus, we have shown that using only the first two moments of the characteristic function, we can estimate our quadratic cost (2.3), and optimize over it in the face of stochasticity regardless of the underlying disturbance process.

Now we reformulate the assumed optimal control model for the expert into an inverse optimal control problem, so we can infer the expert’s control policy.

given $(x_e, u_e)_{[1:N]}$

(3.4a)

$$
\min \mathbb{E}[J_k] = \mathbb{E}[x_{e,k+1}'Qx_{e,k+1} + u_{e,k}'Ru_{e,k}]
$$

(3.4b)

subject to

$$
x[k+1] = Ax[k] + Bu[k] + Dd[k]
$$

(3.4c)

$$
u_{e,k} = K_ex_k
$$

(3.4d)

$$
Q \succ 0, \quad R \succeq 0
$$

(3.4e)

Where our decision variables are now the matrices $Q, R$. We form the Lagrangian from (3.2) and (3.4) to include our constraints,

$$
L(x_{k+1}, u_{e,k}, p_k) = \frac{1}{2}\mathbb{E}[x_{k+1}']Q\mathbb{E}[x_{k+1}] + \frac{1}{2}\mathbb{E}[u_{e,k}']R\mathbb{E}[u_{e,k}]
$$

$$
+ \frac{1}{2}\text{Tr}(Q\Sigma_{x_{k+1}}) + \frac{1}{2}\text{Tr}(R\Sigma_{u_k})
$$

$$
+ p_{1,k}'\mathbb{E}[Ax_k + Bu_{e,k} + w_k - x_{k+1}]
$$

$$
+ p_{2,k}'\mathbb{E}[K_e x_k - u_{e,k}]
$$

(3.5)
where $p_{1,k}$ and $p_{2,k}$ are our dual variables.

From the KKT Conditions, we know that the gradient of our Lagrangian (since we have no constraints on $x$ or $u$) must be zero at our optimal point. Taking the gradients yields:

\[
\nabla_{x_{k+1}} L = Q\mathbb{E}[x_{k+1}] - p_{1,k} = 0 \tag{3.6a}
\]
\[
\nabla_{u_{e,k}} L = R\mathbb{E}[u_{e,k}] + B^T p_{1,k} - p_{2,k} = 0 \tag{3.6b}
\]
\[
\nabla_{p_1} L = A x_k + B u_{e,k} + w_k - x_{k+1} = 0 \tag{3.6c}
\]
\[
\nabla_{p_2} L = K_k x_k - u_{e,k} = 0 \tag{3.6d}
\]

Solving these to find our dual variables we find:

\[
p_{1,k} = Q\mathbb{E}[x_{k+1}] \tag{3.7a}
\]
\[
p_{2,k} = R\mathbb{E}[u_{e,k}] + B^T Q\mathbb{E}[x_{k+1}] \tag{3.7b}
\]

Which simplifies our Lagrangian to:

\[
L(x_{k+1}, u_{e,k}, p_k) = \frac{1}{2} \mathbb{E}[x_{k+1}]^T Q\mathbb{E}[x_{k+1}] + \frac{1}{2} \mathbb{E}[u_{e,k}]^T R\mathbb{E}[u_{e,k}]
+ \frac{1}{2} \text{Tr}(Q\Sigma_{x_{k+1}}) + \frac{1}{2} \text{Tr}(R\Sigma_{u_{e,k}}) \tag{3.8}
\]

Giving us

\[
\text{find } \{Q, R\} \tag{3.9a}
\]
\[
\text{subject to } QA\mathbb{E}[x_k] + B\mathbb{E}[u_{e,k}] + \mathbb{E}[w_k] - p_{1,k} = 0 \tag{3.9b}
\]
\[
R\mathbb{E}[u_{e,k}] - B^T p_{1,k} = 0 \tag{3.9c}
\]
\[
p \succeq 0, \ Q \succeq 0, \ R \succ 0 \tag{3.9d}
\]

for our final inverse problem. Since we have the expert trajectories $(x_e, u_e)$ we can find
the expectations of the state and the input, and since we have complete information about our states, we can compute the expectation of our noise also as:

\[
E[w_k] = D^\dagger (E[x_{k+1}] - A E[x_k] - B E[u_{e,k}])
\]  

with expectations computed via characteristic function and \(D^\dagger\) as the Moore-Penrose pseudoinverse of \(D\).

We condense our proposed method for inferring the expert policy into algorithm 1.

**Algorithm 1** Estimate Feedback Weight Matrices from trajectories of state/input pairs

**Input:** Trajectories \((x_{[1,N]}, u_{[1,N]})_{[1,K]}, K, N \geq 1\)

**Output:** \((Q, R)\)

1. **for** \(n \in [1, N] \) **do** \(\triangleright\) for each state and control action in a trajectory
2. \(\hat{\phi}_x(t) = \sum_{k=1}^{K} \sigma_k(x_n) e^{(jt^\top x_{n,k})} e^{(-\frac{1}{2} t^\top \Sigma t)} \) \(\triangleright\) find ECF of the state
3. \(E[x_n] = \left. \frac{d}{dt} \hat{\phi}_x(t) \right|_{t=0}\)
4. \(E[x_n x_n^\top] = \left. \frac{d}{dt} (\frac{d}{dt} \hat{\phi}_x(t)) \right|_{t=0}\)
5. \(\Sigma_x = E[x_n x_n^\top] - E[x_n] E[x_n]^\top\)
6. \(\hat{\phi}_u(t) = \sum_{k=1}^{K} \sigma_k(u_n) e^{(jt^\top u_{n,k})} e^{(-\frac{1}{2} t^\top \Sigma t)} \) \(\triangleright\) find ECF of the control action
7. \(E[u_n] = \left. \frac{d}{dt} \hat{\phi}_u(t) \right|_{t=0}\)
8. \(E[u_n u_n^\top] = \left. \frac{d}{dt} (\frac{d}{dt} \hat{\phi}_u(t)) \right|_{t=0}\)
9. \(\Sigma_u = E[u_n u_n^\top] - E[u_n] E[u_n]^\top\)
10. \(E[w_n] = D^\dagger (E[x_{n+1}] - A E[x_n] - B E[u_n]) \) \(\triangleright\) Find noise expectation from the trajectory
11. **solve** \((Q_n, R_n)\) output of (3.9) \(\triangleright\) estimate weight matrices for each step
12. **end for**
13. \((Q, R) = (\text{mean}(Q_n), \text{mean}(R_n))\) \(\triangleright\) Find average weight matrices

Algorithm 1 describes calculations performed to solve the inverse optimal control problem (3.9) starting with at least one trajectory, \((x_{[1,N]}, u_{[1,N]})\). For the expert, these are performed offline, with more demonstrations yielding better estimates of the expectations and a more accurate expert policy. The online calculations to perform shared control are in the next section.
3.2 Shared Control

This section describes the methods for computing the optimal shared control policy to control the system from [1].

Using the expert policy estimated previously, we synthesize new automation inputs using a receding-horizon, model-predictive control problem

\[
\min_{t+1} \sum_{k=t}^{t+N} \mathbb{E}[x_{k+1}^T Q x_{k+1} + u_{a,k}^T R u_{a,k}] \tag{3.11a}
\]

subject to

\[
\mathbb{E}[x_{k+1}] = \mathbb{E}[Ax_k + Bu_{a,k} + Dd_k] \tag{3.11b}
\]

with output \(u_{a,k}\) as the automation control input at each time step.

We synthesize our applied control input, \(\hat{u}_k\), as a linear combination of the automation control and the novice’s intended control action:

\[
\hat{u}_k = \theta_k u_{n,k} + (1 - \theta_k) u_{a,k} \tag{3.12}
\]

with \(\theta_k\) as the sharing percentage between the novice and the automation control. \(\theta\) takes discrete values in a finite set, \(Q\), containing values in the interval \([0, 1]\) to determine how finely control can be shared.

Note: when \(\theta = 0\), the automation has full control, and when \(\theta = 1\) the novice is in full control and the automation does not intervene.

The optimal control authority sharing, is determined by solving an optimization problem with a penalty on the automation’s assistance. \(\theta\) optimizes the control
problem,

\[
\min_{\mathcal{K}} \mathbb{E} \left[ \sum_{k=t}^{t+N} (x(k)^T Q x(k) + \hat{u}_k^T R \hat{u}_k + \alpha (1 - \theta)^2 u_{a,k}^T R u_{a,k}) \right] \tag{3.13a}
\]

subject to \[ x_{k+1} = A x_k + B u_{e,k} + w_k \tag{3.13b} \]

\[ \hat{u}_k = \theta u_{e,k} + (1 - \theta) u_{a,k} \tag{3.13c} \]

\[ \theta \in \mathcal{Q} \tag{3.13d} \]

where $\hat{u}_k$ represents the best estimate of the shared input over the interval $k \in [t, t+N]$ at time step $t$. $N$ is a finite horizon and $\alpha \geq 0$ is the tunable penalty on automation intervention. If $\alpha = 0$ then the loss function is always minimized with $\theta = 0$ and the automation takes full control over the system. Larger positive $\alpha$ gives the novice more ability to deviate from the expert without losing control authority [1], but also weakens the automation’s ability to intervene. We can see also that the cost function is lower bounded by $\theta = 0$ and thus the system’s performance is bounded by the automation’s performance regardless of the human’s skill.

During each time step of operation, we solve (3.13) and determine the optimal $\theta \in \mathcal{Q}$. We can obtain finer control by increasing the size of $\mathcal{Q}$ at the cost of computation time.
Chapter 4

Results and Discussion

4.1 Experimental Setup

As described in [1], the 2D simulator, Fig. 4.1, is used as an experimental test bed. Subjects are requested to land the quadrotor slowly on the landing pad while monitoring their progress and the current control authority via the monitor. The state is restricted to state domain $\mathcal{X} \in [-512, 512] \times [0, 768]$ pixels with the starting position uniformly distributed as $x, y \in [-412, 412] \times [568, 668]$ pixels, with all other states starting at zero.

Figure 4.1: 2D landing simulator for data collection and human subject testing
In the original experiments, the system took 1.7 seconds to begin characterizing the novice’s control policy, but the method in this paper is likely to take slightly longer as IOC is slower to find an estimate than the online DMD used in [1]. The expert control policy is pre-computed and loaded into the simulator before run time, and each subject performs 10 total demonstrations after having some time to practice with the simulator before recording begins.

The experiment differs slightly from the simulations, since the initial position will change between trials. To compensate, the ratio of $J_{ratio} = J_e(x_n, u_n)/J_e(x_e, u_e)$ is used for evaluation, with a ratio closer to 1 corresponding to higher performance and a lower ratio indicating poor performance relative to the expert policy.

## 4.2 Initial Data

### 4.2.1 Human Expert

![Human Demonstrations](image)

Figure 4.2: Human demonstrations with mean demonstration used for IRL
We have a human demonstrator who we will presume is acting optimally. Fig. 4.2 shows 30 trajectories and the mean trajectory for our human demonstrations, which we use to infer the expert control policy prior to our simulations.

### 4.2.2 Simulated Novices

Additionally, since we have only this human demonstrator as our expert, we must simulate a novice operator. We chose to simulate a novice as a one-step model predictive control (3.11) with randomized weight matrices $Q_n$ and $R_n$ to account for novice skill being suboptimal. We make the assumption that the novice performs consistently during the trial, but preliminary results indicate that this is not necessarily the case with a real novice.

### 4.3 Simulation Setup

#### 4.3.1 Simulated Plant Dynamics

Data was gathered on a planar quadrotor with state vector

\[
x = [x \ y \ \phi \ \dot{x} \ \dot{y} \ \dot{\phi}]^\top \in \mathbb{R}^6
\]  

(4.1)

control input

\[
\hat{u} = [f_t \ \tau]^\top \in \mathbb{R}^2
\]  

(4.2)

which is applied to the system as

\[
\bar{u} = [\hat{u}^\top \ 1]^\top
\]  

(4.3)

with $f_t$ the total upward force generated by the rotors, and $\tau$ the torque applied around the quadrotor moment of inertia. The appended 1 is a product of removing
the steady-state effects from gravity on the system. We use disturbance vector

$$\mathbf{d} = [f_{wx} \ f_{wy} \ \tau_w]^\top \in \mathbb{R}^3$$  \hfill (4.4)

with $f_{wx}$ and $f_{wy}$ being wind force applied on the x and y axes respectively and $\tau_w$ being torque applied due to the wind. A full derivation of the plant model is included in Appendix A.

### 4.3.2 Simulated Disturbance Models

We performed Monte-Carlo simulations using a variety of noise models to demonstrate that our method performs under a variety of conditions, and we simulate 10 novices performing with 20 iterations of each noise profile. For these simulations we chose $Q = \{0, 0.5, 1\}$ for our set of possible sharing ratios, automation penalty $\alpha = 10$, a starting height between 2.5 m and 3 m, and a starting horizontal offset in $[-2 \text{ m}, 2 \text{ m}]$, [1]. The pdf of each noise profile used is plotted against a standard normal distribution in Fig. 4.3.

Uniform distributions were chosen because they are bounded in an interval with bounds selected directly by the distribution parameters, and are trivially easy to compute.

Similarly, Beta distributions are bounded on the interval $[0, 1]$. They are also common models for bounded real-life disturbance, so demonstrating that the model works with them if of practical value.

We chose two Gamma distributions because they are heavy tailed and have a relatively large non-zero mean. Additionally, the exponential distribution is a special case of the Gamma when the first parameter is 1 which we see in Fig. 4.3c quite clearly.

Much like the gamma, the Laplace distribution is heavy tailed; however, we chose
(a) Uniform distributions used: \( U(-0.1, 0.1) \), \( U(-1, 1) \), \( U(0, 1) \), \( U(0.5, 1.5) \)

(b) Beta distributions used: \( \beta(5, 1) \), \( \beta(1, 5) \), \( \beta(2, 3) \)

(c) Gamma distributions used: \( \Gamma(1, 3) \), \( \Gamma(3, 1) \)

(d) Laplace distributions used: \( \text{Laplace}(0, 1) \), \( \text{Laplace}(0, 2) \)

Figure 4.3: Noise profiles used for simulation

Laplace distributions with zero mean to compare heavy tailed distributions to standard Gaussian noise more clearly. We see that as the scale parameter increases, the distribution becomes flatter and more density is pushed to the tails.

Finally, we included one instance with no noise and two instances with Normal distributed noise \( (N(0, 1) \) and \( N(1, 1) \)) for comparison and to verify that the proposed controller acts correctly with Gaussian disturbance and in the noiseless case.

### 4.4 Simulation Results

In Fig. 4.4, the red line within each bar indicates the median of the data presented, and the red + signs indicate outliers in the data. The mean is indicated by the
colored circles. In Fig. 4.4a, we see that the median and mean rarely line up, except for zero-mean noise profiles where they are relatively close. The median distances for each noise profile indicate that a significant portion of the trials run for each profile were landed successfully or very close to successfully, but there was also large variance in the outcomes, as can be seen with the two Γ’s.

If we take a look at the average distance to the origin for each noise profile, we see that for the bounded disturbance models, the overall performance is considerably better, which is exactly what we expect to see since the set of possible final states is itself a finite set concentrated near the origin in these cases. The vertical dashed lines in Fig. 4.4 shows the division between bounded and unbounded distributions; with the plots in each section sorted by mean, with larger means on the right and smaller means on the left. We see that as the mean value increases, the difficulty in landing safely on the landing pad increases, though the only distributions that did not consistently land safely are the two Γ distributions which have the largest means by a factor of 3. Additionally, the gamma has poor performance compared to other unbounded distributions since all values are positive, so all disturbance samples build additively. Of note, the mean distances in Fig. 4.4a do not consistently follow the median distances, showing that there are some significant outliers in the trajectories.

If we combine the results of Fig. 4.4a with the results in Fig. 4.4b, we can also see that in the zero-mean cases, the average control authority among the novices is much broader and the mean trends towards the median, showing that the novices are given more control in these low-expected-noise environments and do not need as much assistance. While I would like to generalize this to human novices also, part of this is, I suspect, because the simulated novices always act consistently and optimally for their respective weighting matrices; a human on the other hand may attempt to take unanticipated control actions, causing a bigger response than necessary to keep the system under control under some disturbance patterns. For the simulations,
each trajectory begins at the same fixed point to ensure that each noise profile and novice controller got a fair trial. This accounts for the fact that the cost depends on the initial condition, and thus we can’t directly compare cost trajectories when the starting positions differ; this is even more pronounced in the cumulative cost as the difference compounds at each step. The method in simulation deviates slightly in this respect from the experimental setup described above, as in simulation we found clearer results if each novice attempt began from the same initial conditions.

We see in Fig.4.5 that the novices, when uncontrolled, had terrible performance and drifted far from the target landing area. This is reflected in Fig.4.4b where we see that the control authority for all noise profiles hovers around 50% on average, showing that the automation had to intervene consistently. However, we do still see a full range of control authorities for both extremely low noise and high noise cases, which is interesting. In the low noise cases, we expect that the novice can perform reasonably but suboptimally, which is exactly what we see. In the two worst cases, with $\Gamma$ distributed noise, we see that the automation struggled to control the system and the novice sometimes had a better control action. This is because the $\Gamma$ distributions are unbounded and have large positive means, which makes them harder to control. In general, Table 4.1 shows that the proposed method works to move the
shared trajectory towards the optimal trajectory while maintaining control authority above zero, thus accomplishing our goals of letting the novice retain as much authority as possible while bringing the system to more optimal states when the deviations are too large. We also confirm that the system cost is bounded by the expert cost, despite some clearly unstable trajectories in Fig. 4.5, the performance never dips below the expert performance, and we see that the performance of the shared controller brings the mean cost towards the optimal cost while tightening the distribution.

While the quadrotor did not land exactly where the expert did in many trials (Fig. 4.5), it landed within the landing pad during many of them, and so they are classified as safe landings. If the goal was to land it perfectly every time, then it would have made more sense to just use a standard control method [2] and not perform

<table>
<thead>
<tr>
<th>Controller</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novice</td>
<td>6.468e6</td>
<td>2.5753e8</td>
<td>3.199e7</td>
<td>3.339e7</td>
</tr>
<tr>
<td>Expert</td>
<td>4.67e5</td>
<td>1.9484e7</td>
<td>7.595e6</td>
<td>4.192e6</td>
</tr>
<tr>
<td>Shared</td>
<td>1.14e6</td>
<td>1.687e8</td>
<td>1.9378e7</td>
<td>2.2175e7</td>
</tr>
</tbody>
</table>

Table 4.1: Cumulative Cost Statistics

Figure 4.5: Mean trajectory of each noise profile for the expert, unaided novice, and novice aided by shared control
any sharing, but instead we see that it lands 'close enough' that the penalty for automation intervention is more than the cost of being slightly off center. Depending on application, the penalty parameter could be tuned to give the novice less lee-way.

4.5 Future Work

4.5.1 Limitations

It may be noted that the lack of real novice participants could have contributed to our results and conclusions based on those findings. Currently, there are several people training in the use of the quadrotor simulator to become sufficiently skilled as pilots so that we can use them to gather more expert data to refine the inferred optimal control policy, and we are gathering participants to perform as novices to demonstrate the effectiveness of the proposed method.

We also plan to run new experiments with the proposed controller implemented and see how real humans new to the simulator perform. Ideally, the experiments will show similar results to our simulations, but because humans sometimes act unpredictably, it is likely some results will differ.

4.5.2 Expansion to 3D

As a real-life quadrotor is not planar, there is work currently being performed to create a 3-dimensional version of the experimental test bed. This will expand the state from 6 elements to 12 elements, adding a third spatial component, two additional rotational components, and the rates of change of each of these. This allows for a more true-to-life simulation to be performed with the simulator and is the next step toward implementing the controller on a real quadrotor.
Chapter 5

Conclusion

With the growing need to integrate automated control systems with human users, it is becoming more necessary to find solutions to smoothing the interaction between the two. This paper proposed a method for allowing stochastic shared control in the presence of arbitrary disturbances, offering more opportunities for robustness in this growing area. It analyzed the existing literature and current methods for stochastic control, both inverse optimal control and optimal control with Gaussian disturbances, to add a new, more generalized method. This generality will hopefully lead to broader applications of shared controllers and allow them to integrate more smoothly and safely with existing and new cyberphysical systems.

After laying the theoretical groundwork, this paper explored results in simulation for several noise profiles to demonstrate its effectiveness. We hope to explore the efficacy of our method with human subjects in the near future, and believe the proposed controller will perform strongly.
Bibliography


Appendix A

Plant Model Derivation

A.1 Nonlinear Model

We can write the nonlinear dynamics of a planar quadrotor as:

\[
\begin{align*}
\ddot{x} &= \frac{-f_t \sin \phi}{m} \\
\ddot{y} &= \frac{f_t \cos \phi}{m} - g \\
\ddot{\phi} &= \frac{\tau}{J}
\end{align*}
\]  

(A.1)

where \(f_t\) is the total upward force from the propellers, \(\tau\) is the torque around the quadrotor center, \(m\) is the quadrotor mass, \(g\) is acceleration due to gravity, and \(J\) is the quadrotor moment of inertia.

With state vector,

\[
x = [x \ y \ \dot{x} \ \dot{y} \ \phi] \in \mathbb{R}^6
\]  

(A.2)

and control input,

\[
u = [f_t \ \tau] \in \mathbb{R}^2
\]  

(A.3)
we can make a first degree system:

\[
\begin{align*}
\dot{x}_1 &= x_4 \\
\dot{x}_2 &= x_5 \\
\dot{x}_3 &= x_6 \\
\dot{x}_4 &= -\frac{u_1 \sin x_3}{m} \\
\dot{x}_5 &= \frac{u_1 \cos x_3}{m} - g \\
\dot{x}_6 &= \frac{u_2}{J}
\end{align*}
\]  

(A.4)

which we take a small-angle approximation of such that \(\sin(x) \approx x\) and \(\cos(x) \approx 1\).

To remove the steady-state effects due to gravity, we will append a 1 to our input vector,

\[
\bar{u} = [u^\top 1]^\top \in \mathbb{R}^3
\]  

(A.5)

A.2 Linearized Model

Now let us linearize our system around an equilibrium point \(x^*\) with input \(\bar{u}^*\). This solves the algebraic system

\[
\hat{f}(x^*, \bar{u}^*) = 0
\]  

(A.6)

A viable equilibrium point is a hovering state, i.e.,

\[
x^* = [\bar{x} \; \bar{y} \; 0 \; 0 \; 0 \; 0]^\top \in \mathbb{R}^6
\]  

(A.7)

which we will choose to be,

\[
x^* = [0 \; 0 \; 0 \; 0 \; 0 \; 0]^\top
\]  

(A.8)
without loss of generality, with control input

\[ \bar{u}^* = [mg \ 0 \ 1]^T \]  

(A.9)

and we use this to compute the Jacobian of our system.

We find the state transition matrix,

\[ A = \frac{\partial f(x, \bar{u})}{\partial x} \bigg|_{x^*, \bar{u}^*} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]  

(A.10)

We discretize with a sampling time \( \delta t = 0.033s \).

A.3 Disturbance

We add a wind disturbance to our model. Consider the following noise vector:

\[ d = [f_wx \ f_wy \ \tau_w] \in \mathbb{R}^3 \]  

(A.12)
with $f_{wx}$ and $f_{wy}$ being wind force applied on the x and y axes respectively and $\tau_w$ being torque applied due to the wind. Since these are forces, they don’t apply directly to our state variables, but through the following relation:

$$D = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\frac{1}{m} & 0 & 0 \\
0 & \frac{1}{m} & 0 \\
0 & 0 & \frac{1}{J}
\end{bmatrix}$$

(A.13)

Note: When we implement noise matrix $D$, it is recommended to scale it with sample time $\delta t$ for more accurate performance.
A.4 Final State Space Model

Our final state space model used is:

\[
x[k + 1] = \begin{bmatrix}
1 & 0 & 0 & \delta t & 0 & 0 \\
0 & 1 & 0 & 0 & \delta t & 0 \\
0 & 0 & 1 & 0 & 0 & \delta t \\
0 & 0 & -g\delta t & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
x[k] + \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & \frac{\delta t}{m} & 0 \\
0 & \frac{\delta t}{J} & 0
\end{bmatrix} \dot{u}[k]
\]

\[+ \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & \frac{\delta t}{m} & 0 \\
0 & \frac{\delta t}{m} & 0 \\
0 & 0 & \frac{\delta t}{J}
\end{bmatrix} d[k]
\]

with values

<table>
<thead>
<tr>
<th></th>
<th>Quadrotor mass</th>
<th>0.18Kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>Gravitational acceleration</td>
<td>9.8m/s²</td>
</tr>
<tr>
<td>J</td>
<td>Quadrotor moment of inertia</td>
<td>(2.5 \times 10^{-4})Kgm²</td>
</tr>
</tbody>
</table>

Table A.1: Quadrotor constants