2020

FAST DECISION-MAKING UNDER TIME AND RESOURCE CONSTRAINTS

Kyle Gabriel Lassak
West Virginia University, klassak@mix.wvu.edu

FAST DECISION-MAKING UNDER TIME AND RESOURCE CONSTRAINTS

Kyle Gabriel Lassak

**Dissertation submitted to the Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of**

**Doctor of Philosophy
in
Mechanical Engineering**

**Yu Gu, PhD, Committee Chairperson
Jason Gross, PhD
Patrick Browning, PhD
Natalia Schmid, PhD
Powsiri Klinkhachorn, PhD**

Department of Mechanical and Aerospace Engineering

**Morgantown, West Virginia
2020**

Keywords: Continuous Partially Observable Markov Decision Process (POMDP), Iterative Linear Quadratic Gaussian (iLQG), Outfielder Problem, Catching Heuristics, Target Interception.

Abstract

FAST DECISION-MAKING UNDER TIME AND RESOURCE CONSTRAINTS

by Kyle G. Lassak

Practical decision makers are inherently limited by computational and memory resources as well as the time available in which to make decisions. To cope with these limitations, humans actively seek methods which limit their resource demands by exploiting structure within the environment and exploiting a coupling between their sensing and actuation to form heuristics for fast decision-making. To date, such behavior has not been replicated in artificial agents. This research explores how heuristics may be incorporated into the decision-making process to quickly make high-quality decisions through the analysis of a prominent case study: the outfielder problem. In the outfielder problem, a fielder is required to intercept balls traveling in ballistic trajectories, while the motion of the fielder is constrained to the ground plane. In order to maximize the probability of interception, the agent must make good, yet timely, decisions. Researchers have put forth several heuristic approaches to describe how a fielder may decide how to run based only on immediately available information under different control paradigms. This research statistically quantifies upper bounds on the expected catch rate of a couple notable approaches, given that interception of the ball is theoretically possible if the fielder ran directly towards the landing spot with maximal effort throughout the entire duration of the ball's flight.

Additionally, novel modifications are made to a belief-space variant of iterative Linear Quadratic Gaussian (iLQG), which is an online method that may be used to find locally-optimal policies to continuous Partially Observable Markov Decision Processes

(POMDPs) in which Bayesian estimation may reasonably be approximated by an Extended Kalman Filter (EKF). Directional derivatives are used to reduce the computation time of certain matrix derivatives with respect to the variance of the belief state from $O[n^4]$ to $O[n^3]$, where n is the dimension of the belief space. However, the improved algorithm still may not be capable of real-time decision-making by the standards of modern-day computing on mobile platforms, especially in systems with long planning horizons and sparse rewards. The belief-space variant of iLQG is applied to the outfielder problem, which may also indicate its applicability to similar target interception problems with input constraints, such as missile defense.

ACKNOWLEDGEMENTS

I would like to first thank my advisor, Dr. Yu Gu, for his unwavering support and counsel in matters of research and life. I could not have asked for a better mentor. Thank you for all you have done for me.

Next, I would like to thank the rest of my committee for their openness in discussing ideas and their willingness to support me on my academic journey.

I also owe a great deal of thanks to the many current and former members of the West Virginia University Interactive Robotics Laboratory, Flight Control Systems Laboratory, Navigation Lab, and Centennial Challenge teams for your help and your friendship. You helped make this pursuit both enlightening and enjoyable, and for that you are very much appreciated.

Finally, I would like to thank my parents and my family: Jessie, Annabelle, and Zander. You have sacrificed the most of all to help get me through this, and I can never thank you enough for your love and support. You could not carry it for me, but you did carry me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Acronyms

DOF	Degree of Freedom
EKF	Extended Kalman Filter
GAG	Gaining Angle of Gaze
GOAC	Generalized Optical Acceleration Cancellation
iLQG	Iterative Linear Quadratic Gaussian
LQG	Linear Quadratic Gaussian
LOT	Linear Optical Trajectory
MDP	Markov Decision Process
OAC	Optical Acceleration Cancellation
PID	Proportional-Integral-Derivative
POMDP	Partially Observable Markov Decision Process
UKF	Unscented Kalman Filter

English Symbols[1]

A	Generic matrix
$\mathcal{A}$	Set of actions
a	An action
$\mathcal{B}$	Set of belief states
b	Belief state
c	Cost function
$\boldsymbol{d}$	Relative distance vector
$\boldsymbol{f}$	State dynamics
g	Gravity
$\boldsymbol{h}$	Measurement function
I	Identity matrix
K	Kalman gain
k	Discrete time index
ℓ	Length of the horizon
M	Process noise covariance
$\boldsymbol{m}$	Process noise
N	Measurement noise covariance
$\boldsymbol{n}$	Measurement noise
$\mathcal{O}$	Observation function for partially observable Markov decision process
o	An observation
P	*A posteriori* covariance of the state estimate
$\mathcal{R}$	Reward function
$\boldsymbol{r}$	Relative image distance between the ball and home plate

[1] Many symbols used in this work cannot be described concisely here, however, their definitions are stated within the text.

$\mathcal{S}$	Set of states
S	Quadratic term in the value function approximation
s	Constant term in the value function approximation
$\boldsymbol{s}$	Linear term in the value function approximation
T	Belief variance weighting in the value function approximation
$\mathcal{T}$	Transition function for (partially observable) Markov decision process
t	Continuous time
$\boldsymbol{u}$	Input
W	Covariance of belief update
$\mathcal{W}$	Weighting matrix
$\boldsymbol{w}$	Deviation due to belief update
x	The x-component in Cartesian coordinates
$\boldsymbol{x}$	State vector
y	The y-component in Cartesian coordinates
$\boldsymbol{y}$	Measurement vector
z	Generic scalar; or the z-component in Cartesian coordinates
$\boldsymbol{z}$	Generic vector

Greek Symbols[2]

α	Angle of elevation of the ball with respect to the fielder; or a scaling factor
$\boldsymbol{\alpha}$	An α-vector: vector normal to a hyperplane in belief space
β	Lateral angle formed between the ball, fielder, and home plate
Γ	*A priori* covariance of the state estimate
$\mathit{\Gamma}$	A set of α-vectors
γ	Angle between relative image position and velocity
ϵ	Catch radius
θ	Fielder's heading
ϑ	Launch angle of the ball
Λ	Diagonal matrix
π	A policy; or a famous irrational number
ρ	Reward function of a belief Markov decision process
σ	Standard deviation
τ	Transition function of a belief Markov decision process
υ	Value function
φ	Lateral angle of the ball's initial direction
ψ	Angle between the horizon and vector from home plate to the ball in the image
Ω	Set of observations

[2] Many symbols used in this work cannot be described concisely here, however, their definitions are stated within the text.

Notation[3]

A	A matrix variable
z	A scalar variable
$\boldsymbol{z}$	A vector variable
$A[\cdot]$	A matrix function
$z[\cdot]$	A scalar function
$\boldsymbol{z}[\cdot]$	A vector function
$\mathrm{tr}[\cdot]$	Matrix trace
$\mathrm{vec}[\cdot]$	A column vector formed by concatenating the columns of a matrix
$\mathrm{diag}[\cdot]$	A diagonal matrix with the elements of the argument along the diagonal
$O[\cdot]$	Order of the complexity
$\mathrm{E}[\cdot]$	Expected value of a random variable
$\mathrm{Var}[\cdot]$	Covariance of a multivariate random variable
$\Pr[\cdot]$	Probability of an event
$a\|b$	a is conditioned on b
$\max_a$	Maximization over a
$\arg\max_a$	Returns the argument which maximizes a given function
$\langle\cdot\rangle$	A tuple
$\|\cdot\|$	Absolute value
$\|\|\cdot\|\|$	Euclidian norm
$\mathcal{N}[\boldsymbol{\mu}, P]$	Multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance P
Δz	A perturbation of variable z
δP	A matrix that is an arbitrary scalar multiple of
$A > 0$	A is a positive-definite matrix variable
$A \geq 0$	A is a positive semi-definite matrix variable

Accent Characters

$\hat{z}$	The mean estimate of a random variable z
$\bar{z}$	The nominal value of z
$\dot{z}$	First derivative of z with respect to time
$\ddot{z}$	Second derivative of z with respect to time

Superscripts

$*$	The optimal value
(i)	The value at iteration i
T	Matrix transpose
-1	Matrix inverse

[3] Common conventions of trigonometry, linear algebra, and matrix calculus also apply.

Subscripts

i	The value of a vector at index i
b	A property belonging to the ball
f	A property belonging to the fielder
h	A property belonging to home plate
k	The value at discrete time k
t	The value at continuous time t
u	The u-component in image coordinates
v	The v-component in image coordinates
x	The x-component in Cartesian coordinates
y	The y-component in Cartesian coordinates
z	The z-component in Cartesian coordinates

1 INTRODUCTION

From the moment that a fly ball is hit, a baseball fielder has only a few seconds to run to the spot where the ball will land in order to catch it. The task of fielders to position themselves at the correct spot at the time the ball lands (neglecting the manipulation task of actually catching the ball) is commonly referred to as the *outfielder problem* [32], and has drawn increasing interest from a variety of researchers in the fields of cognitive science, artificial intelligence, and robotics. It draws attention because humans' proficiency in performing such a complex and time-sensitive task seems to be at odds with their limitations in sensing, working memory, time estimation, and computation power (at least conscious and effortful computational power), to name a few. These limitations prohibit humans from determining the optimal running paths in real-time, e.g. by solving for the optimal solution of the Partially Observable Markov Decision Process (POMDP [43]) which would theoretically give them the highest probability of catching the ball.

Therefore, it has been proposed that humans must rely on *heuristic* approaches in order to arrive at good decisions in a timely manner. Heuristics are defined by Pearl [74] as "strategies using readily accessible though loosely applicable information to control problem-solving processes" that also "represent compromises between two requirements: the need to make such criteria simple and, at the same time, the desire to see them discriminate correctly between good and bad choices." Researchers have posited several different heuristic methods that humans may implement in the outfielder problem, with some experimental data showing that human running trajectories are often similar to those expected by the proposed heuristics. However, no method has yet demonstrated the

full capacity of human fielding behaviors (particularly humans' predictive abilities; [11]), and a satisfactory description of how humans resolve the outfielder problem remains an open question.

While the study of these heuristics provides significant value to the understanding of human and animal behavior [92], as well as possible strategies for robotic implementations (e.g. [107]), they are perhaps just an instantiation of a broader real-time decision-making strategy for resource-limited agents. One plausible explanation for the success of humans in the outfielder problem that is given by researchers in the cognitive sciences is the theory of *embodied cognition* [6][123]. Embodied cognition is a broad area of study emphasizing the role of the motor system, the perceptual system, and bodily interactions with environment in complementing the cognitive process. Specifically, it has been hypothesized that a human fielder employs embodied cognition in the outfielder problem to select actions that leverage the fielder's interaction with the environment to reduce the amount of computation that needs to be performed by the fielder, therefore enabling the fielder to quickly decide how to act [124].

The POMDP framework provides a rich theoretical foundation for decision-making in stochastic control problems, where the value of an action may be interpreted as a trade-off between receiving a direct reward and acquiring information about the state of the environment [43]. However, finding optimal solutions to POMDPs is computationally intractable [43], therefore researchers seeking approximate solutions to POMDPs (e.g. [43][50][80][112]) must necessarily factor resource constraints into their methodology. However, algorithms do not exist which autonomously optimize decision-making strategies to an agent's resource constraints in a manner similar to human abilities.

Generally, it is the job of the human researcher to select, modify, or design an algorithm which fits the specific space and time constraints of an agent in a given system, whereas humans seem to actively seek efficient methods which suit their specific machinery (whether this occurs consciously or subconsciously, see Section 7.2).

One algorithm that exists in literature for efficiently finding locally-optimal policies to continuous POMDPs in which Bayesian estimation may reasonably be approximated by an Extended Kalman Filter (EKF; [97][121]) is the belief space variant of iterative Linear Quadratic Gaussian (iLQG) that was proposed by van den Berg et al. [118]. In their work, van den Berg et al. [118] developed an algorithm which iteratively improved on a linear nominal policy which would converge to a locally optimal policy with a second-order convergence rate, while the complexity of a single iteration was $O[\ell n^4]$, where n is the number of states and ℓ is the length of the planning horizon. Furthermore, it is proposed in this work that directional derivatives can be employed to calculate certain matrix derivatives with respect to the variance of the belief state, which reduces the time complexity of determining those derivatives from $O[n^4]$ to $O[n^3]$. The determination of these derivatives forms a computational bottleneck under certain conditions (i.e. linear dynamics and low-dimensional action and observation spaces). Therefore, the run-time of the algorithm is always improved while the efficiency of the algorithm may be improved by up to an order of magnitude under special conditions.

However, despite the improvements to the efficiency of the iLQG algorithm, there still exist several limitations which complicate its direct application to the outfielder problem. For instance, the iLQG algorithm assumes a value function that is convex with respect to the parameters of Gaussian beliefs (i.e. mean and variance), which is not true in

general. Some of these limitations were mitigated through the use of cost shaping, however significant assumptions were employed to make the algorithm operable which were undesirable (e.g. the maximum likelihood assumption about the ball's time-to-impact was employed). Additionally, the running time of the modified belief iLQG until convergence is still too long for it to be employed in real time, which highlights the need to find alternate methods to quickly find good decisions.

Sports provide relatable and tangible examples in which resource-limited agents (e.g. humans) must make time-critical decisions, however there are innumerable examples in which artificial agents similarly must make time-critical decisions using limited resources. For example, closely related to the outfielder problem are other target interception problems, such as missile defense [126]. While the dynamics, actuation, and sensing of a missile defense system may be quite different from agents that are used to catch baseballs, the underlying principle that the coupling between actuation and sensing can be exploited to reduce the computational effort of the agent remains similar. The methodologies which allow the belief iLQG algorithm to be applied to the outfielder problem thus may also be applicable to missile defense. It has also been proposed that robots can throw and catch objects for the efficient transportation of small objects [34], which is essentially the same motive for why throwing and catching is performed in baseball and other sports. Additionally, similar principles may be applied to improve the autonomy of vehicles such as cars and planetary rovers. Cars require fast decision-making to cope with unexpected situations while traveling at high speeds [20], while planetary rovers require efficient decision making strategies because they are notably deficient in computational resources [9].

1.1 OBJECTIVE

The objective of this dissertation is to present novel modifications which improve the efficiency of a belief space variant of iLQG presented by van den Berg et al. [118], and to evaluate several control methods – including the modified iLQG algorithm – which may be used to resolve the outfielder problem.

The belief space variant of iLQG presented by van den Berg et al. [118] was improved through the implementation of directional derivatives [106]. The directional derivative specifies the rate of change of a multivariate function in the direction that is specified by a unit vector. In this context, directional derivatives are exploited to efficiently calculate certain first-order Taylor series approximations to reduce the time complexity of the belief space variant of iLQG presented by van den Berg et al. [118], while the underlying functionality of the algorithm remains unchanged. In effect, these modifications reduce the computational bottleneck of a single iteration of the algorithm from $O[n^4]$ to $O[n^3]$, although the full benefits are only realized under special circumstances.

This modified belief iLQG is compared with various catching heuristic approaches found in literature (e.g. [21][56][61][114]). The actions are optimized for each heuristic method, so that an upper bound on the expected catch rate of each heuristic method can be approximated by simulating a large number of trials in different noise configurations. Furthermore, this work explores how heuristic techniques fit into the larger framework of existing POMDP research and provides insight into further research in this area.

1.2 ORGANIZATION

This rest of this dissertation proceeds as follows. Chapter 2 explores the theoretical background of optimal decision-making in uncertain environments and the methodologies employed in finding exact and approximate solutions. Chapter 3 provides a synopsis of existing research into the outfielder problem, and it also provides some connections to general practical decision-making in uncertain environments. Chapter 4 provides novel modifications to the belief space variant of iLQG presented by van den Berg et al. [118]. Chapter 5 describes the modeling of the fielder and the ball for the outfielder problem that was studied in this work, which also provides some improvements to the fielder's measurement and noise models that had been considered in previous work (e.g. [11][38]). Chapter 6 describes the predictive methods that were used to resolve outfielder problem, including deterministic time-optimal control and the modified belief iLQG controller described in Chapter 4. Chapter 7 describes how various fielding heuristics were implemented so that the upper bound on the expected performance of the heuristic strategies could be approximated. Chapter 8 provides simulated results of the controllers described in Chapters 6 and 7 under various noise configurations. Chapter 9 provides concluding remarks about the methodologies employed and the challenges encountered in this work, and the direction of future research.

2 THEORETICAL BACKGROUND

Decision-making in partially observable domains has received extensive research since Sondik's [101] seminal work. Optimal decision-making requires consideration of the value of future information that can be acquired by executing an action, the expected

future reward of an action, and how a given action changes the environment [43]. These considerations are summarized by the agent's value function, which is introduced first through the consideration of Markov decision processes.

2.1 MARKOV DECISION PROCESSES

A Markov Decision Process (MDP; [109]) is a representation of an environment in which an agent makes decisions in discrete time. The state of the environment is fully observable to the agent at each time step, however each action performed by the agent results in an uncertain transition. Additionally, an MDP exhibits the Markov property, which implies that the history does not provide any more information about future states of the system than the current state. Formally, an MDP is given by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where

- $\mathcal{S}$ is a set of states, $s \in \mathcal{S}$
- $\mathcal{A}$ is a set of actions, $a \in \mathcal{A}$
- $\mathcal{T}[s, s', a] = \Pr[s_{k+1} = s' | s_k = s, a_k = a]$ is the transition function. It is the conditional probability that the state will transition from state s to state s' if action a is implemented.
- $\mathcal{R}_k[s, a]$ is the reward function. It describes the expected immediate reward received by the agent for executing action a at state s at time k.

A policy $\pi_k[s]$ is a mapping from a state to an action at time step k. If a policy is the same at all time steps, i.e. $\pi_k[s] = \pi[s]\ \forall k$, then the policy is a *stationary policy*. If, however, the policy changes based on the time index, then the policy is *nonstationary* [109].

The optimal state-value function $v_k^*[s]$ is the total expected reward the agent will receive given that the agent begins in state $s_k = s$ at time k and executes the optimal action $a_k = a^*$ at each time step until the end of the trial, where the optimal action a^* is defined as the action which maximizes the total expected future reward given that the agent also acts optimally in future time steps. Bellman's equation [109] states that

$$\begin{aligned} v_k^*[s] &= \max_a \left[E[\mathcal{R}_k[s_k, a_k] + v_{k+1}^*[s_{k+1}] \mid s_k = s, a_k = a] \right] \\ &= \max_a \left[\mathcal{R}_k[s, a] + E\left[v_{k+1}^*[s'] \right] \right] \\ &= \max_a \left[\mathcal{R}_k[s, a] + \sum_{s'} \mathcal{T}[s, s', a]\, v_{k+1}^*[s'] \right] \end{aligned} \tag{1}$$

In words, Bellman's equation states that $v_k^*[s]$ is equal to the expected immediate reward for executing optimal action a^* in state s at time k plus the expected value of the state-value function over all possible terminal states s'. A discount factor $\gamma \in [0,1]$ is often multiplied to the second term in Equation 1, but is omitted in this work since only finite horizon problems are considered. The optimal policy $\pi_k^*[s] = a^*$ is the mapping from state s to the optimal action a^*, which is the argument of Equation 1.

$$\pi_k^*[s] = \arg\max_a \left[\mathcal{R}_k[s, a] + E\left[v_{k+1}^*[s'] \right] \right] \tag{2}$$

The goal of most problems which invoke MDPs is to find an optimal policy $\pi_k^*[s]$ which maximizes the value of the state-value function, i.e. the policy which maximizes the expected total future reward. One algorithm which is guaranteed to converge to the optimal policy is *value iteration* [109]. In this work, only finite horizon problems are considered, since the ball is anticipated to impact the ground in a finite time, with ℓ being the length of the horizon. Additionally, it is assumed that the reward function at the final

time step $\mathcal{R}_\ell[s]$ is only a function of the state because no further actions will be taken. Value iteration may be initialized by setting the value function at time step ℓ to be equal to the reward function at time step ℓ.

$$v_\ell[s] = \mathcal{R}_\ell[s] \tag{3}$$

Then, the value of the value function at each state s and time step k is updated by acting greedily with respect to the value function in the next time step.

$$v_k[s] = \max_a \left[\mathcal{R}_k[s,a] + E\left[v_{k+1}[s']\right]\right] \tag{4}$$

Equation 4 is often referred to as the Bellman update [109]. Here, finite-horizon problems which may require nonstationary policies are considered, since generally the optimal policy of a finite-horizon MDPs is nonstationary [43]. *Policy iteration* is another method for finding optimal policies and is similar to value iteration. Instead of inferring the policy from the value function, policy iteration begins by assuming an initial policy and calculating the value of the initial policy. Then, the policy is updated by acting greedily with respect to the value function given by the current policy (similar to Equation 4).

Finding good policies in large MDPs – ones which are too large to efficiently obtain exact solutions by value or policy iteration – is one of the core problems in reinforcement learning. While these MDPs are too large to find exact solutions, there exist numerous algorithms which can provide good approximate solutions. These algorithms often rely on value or policy iteration as an essential component of the approximation strategy [109].

2.2 PARTIALLY OBSERVABLE MARKOV DECISION PROCESSES

A Partially Observable Markov Decisions Process (POMDP; [43]) is an MDP in which the agent cannot fully observe the state of the environment, which dramatically complicates task of finding an optimal policy. A POMDP may formally be represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{R}, \mathcal{O} \rangle$:

- $\mathcal{S}$ is a set of states, $s \in \mathcal{S}$
- $\mathcal{A}$ is a set of actions, $a \in \mathcal{A}$
- Ω is a set of observations, $o \in \Omega$
- $\mathcal{T}[s, s', a] = \Pr[s_{k+1} = s' | s_k = s, a_k = a]$ is the transition function. It is the conditional probability that the state will transition from state s to state s' if action a is implemented.
- $\mathcal{R}_k[s, a]$ is the reward function. It describes the expected immediate reward received by the agent for executing action a at state s and time step k.
- $\mathcal{O}[s', o, a] = \Pr[o_{k+1} = o | s_{k+1} = s', a_k = a]$ is an observation function, which gives the conditional probability of making an observation o given the agent took action a and ended up in state s'.

Again, an optional discount factor γ is omitted in this work because only finite horizon problems are considered.

A POMDP can be reduced to an MDP known as a *belief MDP* through the introduction of the belief state b. The *belief state* is the probability distribution over the environmental states $\mathcal{S}$ given the agent's *history* – the set of all recorded actions and observations in a trial – and $b[s]$ denotes the probability of being in environmental state s, i.e. $b[s] = \Pr[s]$. Due to the Markov property, b represents all the useful information

from the agent's history. The new belief state b', given an old belief b, an action a, and an observation o, is computed via a Bayesian estimator:

$$\begin{aligned} b'[s'] &= \Pr[s'|b,a,o] \\ &= \frac{\Pr[o|\, s',a,b]\,\Pr[s'|a,b]}{\Pr[o|\, a,b]} \\ &= \frac{\Pr[o|\, s',a]\sum_{s\in\mathcal{S}}\Pr[s'|s,a,b]\,\Pr[s|a,b]}{\Pr[o|\, a,b]} \\ &= \frac{\mathcal{O}[s',o,a]\sum_{s\in\mathcal{S}}\mathcal{T}[s,s',a]\,b[s]}{\Pr[o|\, a,b]} \\ &= \mathrm{SE}[b,a,o] \end{aligned} \tag{5}$$

where Equation 5 can be obtained from the application of Bayes' Theorem, the Markov property, the law of total probability for the second term in the numerator, and from the substitution of the observation and transition functions. The updated belief state b' can thus be found using the state-estimation function $\mathrm{SE}[b,a,o]$, which calculates the probability of each environmental state of a given the belief state using Equation 5.

A belief MDP defined as a tuple $\langle \mathcal{B}, \mathcal{A}, \tau, \rho \rangle$:

- $\mathcal{B}$ is a set of belief states, $b \in \mathcal{B}$
- $\mathcal{A}$ is a set of actions, $a \in \mathcal{A}$
- $\tau[b,b',a] = \Pr[b_{k+1} = b' \,|b_k = b, a_k = a]$ is the belief state transition function. It is dependent upon the probability of receiving an observation o given a belief b and an action a executed at that belief:
- $\Pr[b' \,|b,a] = \sum_{o\in\Omega}\Pr[b' \,|b,a,o]\,\Pr[o|\, b,a]$

  where

$$\Pr[b' \,|b,a,o] = \begin{cases} 1 & \text{if } \mathrm{SE}[b,a,o] = b' \\ 0 & \text{otherwise.} \end{cases}$$

The belief state transition function is the conditional probability that the belief state will transition from belief state b to belief state b' if action a is implemented.

- $\rho_k[b,a] = \mathrm{E}_b\left[\mathcal{R}_k[s,a]\right] = \sum_{s\in\mathcal{S}} b[s]\mathcal{R}_k[s,a]$ is the reward function. It describes the immediate reward received when action a is implemented at belief state b. It is equal to the expected reward of the environmental states found by using the distribution of the belief state.

The belief transition function defines a conditional probability distribution over belief states, i.e. a probability distribution over probability distributions. This arises due to the fact that there is a probability that one of several observations will be made after action a is implemented at belief state b, and each possible observation will generate a different value of the belief state as calculated by the state estimator. However, after a particular observation o is made, the belief state b' – as calculated by the state estimator – is unique [43]. Therefore, the belief state is fully observable, so the belief MDP is an MDP operating on the belief state rather than the environmental state.

2.2.1 METHODS FOR SOLVING DISCRETE POMDPS

The reduction of POMDP problems to belief MDPs enables the tools which can be used to solve MDPs to be applied to solve POMDPs, namely value and policy iteration. Some additional considerations are necessary due to the fact that the belief space is an infinite dimensional continuous space, necessitating that the value function and the policy must be defined over an infinite dimensional belief space. While solutions to continuous space MDPs are generally difficult to find due to the large state space, the value functions of belief MDPs possess additional structure which renders it possible to find exact solutions with finite representations. Specifically, the optimal value function of

a POMDP with a finite time horizon k is piecewise linear and convex [43]. This means that the optimal value function can be defined as the maximum value of the set Γ_k consisting of $|\mathcal{S}|$-dimensional hyperplanes defined over belief space. These hyperplanes are usually each represented by an α-vector – which is the normal vector of the associated hyperplane – so that $\Gamma_k = \{\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_m\}$, where m is the number of hyperplanes that are necessary to define the optimal value function. The k-step optimal value function V_k can then be represented as

$$V_k[b] = \max_{\boldsymbol{\alpha} \in \Gamma_k} \sum_{s \in \mathcal{S}} b[s]\boldsymbol{\alpha}[s] \tag{6}$$

where $\boldsymbol{\alpha}[s]$ indicates the magnitude of $\boldsymbol{\alpha}$ in the direction associated with state s.

Exact algorithms solve for the complete set of α-vectors which compose V_k. This would also implicitly define the optimal policy. However, finding exact solutions is often difficult due to the computational complexity of the solution: finite-horizon POMDPs are PSPACE-complete while infinite-horizon POMDPs are undecidable [86]. Therefore, finding efficient approximations to POMDP solutions is the focus of most POMDP research.

2.2.1.1 Offline Methods

Grid-based methods provide an intuitive approach to approximating the value function over the belief space. In these methods, the belief space is discretized, and the value function is approximated at specific points on the grid. Largely, these methods differ based on the method in which grid points are generated (e.g. fixed or variable grid size) and how the value function is approximated at the grid points and then generalized to the entire belief space [14][15][127]. Grid-based methods have largely fallen out of

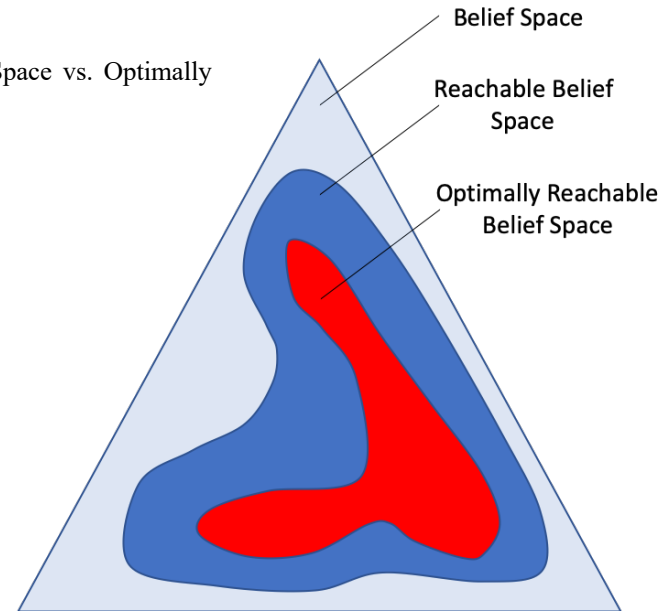


Figure 1: Belief Space vs. Reachable Belief Space vs. Optimally Reachable Belief Space.

favor because they are sensitive to the curse of dimensionality – as the number of states grows, the number of grid points that are necessary to discretize the belief space sufficiently enough for a good solution grows exponentially, even for variable size grids.

Point-based methods [50][77][99][102] cope with the curse of dimensionality by sampling points from the belief space, and approximating the value function at these sample beliefs. Sampling can be made more efficient by considering only the *reachable* belief space – the subset of belief space that is reachable given an initial belief state and any given set of actions – rather than by trying to approximate the value function over the whole belief space [77]. Search heuristics may also be employed to focus on only the *optimally reachable* belief space – the subset of belief space that is reachable given an initial belief state and optimal actions [50]. Like grid-based methods, point-based methods differ in how beliefs are sampled and the manner in which the value function is approximated at the sample beliefs. Since point-based methods typically focus only the reachable belief space, the value function approximations that they generate can only be

generalized over a subset of the belief space that is reachable (or optimally reachable) from some initial belief state. If the POMDP is initialized with a different initial belief where the value function is not well suited, a new approximation of the value function may be necessary. Point-based methods have been shown to be effective in relatively large discrete POMDPs, e.g. an integrated exploration problem with $|\mathcal{S}| = 15{,}517$, $|\mathcal{A}| = 8$, and $|\Omega| = 1{,}015$ has been efficiently approximated using SARSOP [50]. In general, the effectiveness of point-based algorithms is likely governed by the complexity of the reachable belief space [40]. Many point-based algorithms enable a trade-off between the complexity of the algorithm and the quality of the approximation by varying the number of sample points. However, point-based algorithms are still subject to the curse of dimensionality, which requires that the number of sample points that are necessary for good performance to grow exponentially with the dimensionality of the belief space [99].

Another method for mitigating the curse of dimensionality is through belief compression. Compression methods [82][87] map the belief space to some lower dimensional compressed belief space. A compression is considered to be *lossless* if the compressed belief state can be used to accurately evaluate the value of all policies, otherwise the compression is considered to be *lossy* [82]. Lossless compressions that result in an appreciable reduction in the dimensionality of the belief space are often intractable to find or may not exist, so it is often desirable to find lossy compressions that minimize the error between the value of a policy that is evaluated on the compressed POMDP versus the value of the policy that is evaluated on the original POMDP. Compression algorithms can thus be used as a preprocessing step to point-based methods

in order to reduce the dimension of the belief space for which the POMDP must be solved [102].

Value iteration methods implicitly represent a policy as the action at each belief state that maximizes the total expected future reward, which is given by the value function. In contrast, policy-based methods [1][37][69][83] iteratively attempt to improve an explicitly defined policy. For example, finite state controllers have been used to directly map beliefs into actions [1][37]. Finite state controllers are appealing because the value function of a finite state controller is piecewise linear and convex [37], and it is also easy to evaluate. Additionally, a finite state controller can be expanded in size so that the error between the finite state controller and the optimal policy can be made arbitrarily small. However, due to time and memory constraints, it is necessary to limit the size of the finite state controller to a fixed finite value. Finding the optimal finite state controller of a fixed size is NP-hard [37], so only locally optimal solutions are obtainable. Some search heuristics allow moderate expansion of the finite state controller to escape local optima; however, they still do not enable the globally optimal fixed-size finite state controller to be found efficiently.

It has been also been observed that it is sometimes easier to improve upon a given policy than it is to determine than the value of a policy [1]. This may occur when a relatively simple finite state controller yields good performance while the corresponding value function is complex. To avoid evaluating the value of a policy, gradient based methods can be used, which only require that the direction in policy space which maximizes the average reward to be computed [1]. Thus, the policy can be improved by stepping in the direction of the gradient without having to evaluate the policy.

2.2.1.2 Online Methods

In large POMDPs, offline methods can practically only provide coarse approximations of the value function, which often causes the resulting policy to be of poor quality. Therefore, it is often necessary to employ an online method which searches for the best action for the current belief state only, rather than trying to find the optimal policy over the whole belief space [86]. This may be accomplished through forward search on the tree of all possible future action-observation sequences with the current belief state at the root node. Forward search is used to locally approximate the value of each action at the current belief state, so that the agent then immediately executes the action with the maximum expected total reward. Fully expanding the tree is intractable unless the planning horizon is short and the POMDP contains small action and observation spaces. Branch and bound pruning, Monte Carlo methods, and heuristic search have been employed to reduce the necessary expansion of the action-observation tree when these conditions are not satisfied. These methods comprise a list of online techniques that is neither mutually exclusive nor exhaustive, but have demonstrated to be effective in some large domains [86].

Branch and bound pruning (e.g. [72]) involves creating upper and lower bounds for the value of each action at the fringe nodes of the tree. If the upper bound of the value of one action is less than the lower bound of another action, then the branch of the tree that descends from that action is pruned, which avoids superfluous expansion of descendent nodes. Finding efficient ways of establishing bounds is therefore essential for branch and bound pruning methods to be implementable.

Monte Carlo methods (e.g. [96]) involve simulating many possible histories. Rather than searching the full breadth of possible observations, histories are sampled at depth. This means that longer sequences of actions and observations are sampled rather than considering several possibilities at each time step. Monte Carlo methods provide the benefit of a reduced branching factor, so planning can be done further into the future. However, since the full breadth of possibilities is not considered, it is possible branches with high expected total reward are missed in planning.

Heuristics search techniques (e.g. [99]) are used to selectively expand the tree at the node that the heuristic predicts will provide the greatest improvement to the solution. This mitigates the problem of the branching factor while also focusing on branches with high expected total reward. However, these methods may run slowly if there are many nodes to consider or if evaluating the heuristic is expensive.

Online algorithms are often used in conjunction with an offline algorithm, which is used to provide a coarse approximation of the value function at fringe nodes. Better approximations of value function may also be learned concurrently with online planning, so that the approximation of the value function is continuously improved whenever online planning is executed [86].

2.2.2 METHODS FOR CONTINUOUS POMDPS

Continuous POMDPs are POMDPs with continuous state spaces and are analogous to discrete POMDPs. Continuous state spaces present a unique set of challenges and opportunities that differentiate many continuous POMDP methods from their discrete counterparts. The challenges stem from the fact that the dimensionality of

the state space in a continuous POMDP is uncountably infinite [17], which in turn causes the number of continuous dimensions of the belief space to be uncountably infinite.

While state-estimation is usually an easy task in the discrete state spaces considered by most discrete methods (with some exceptions, e.g. [96]), even representing the belief state is often intractable in continuous spaces. This is because beliefs must be represented by a finite set of parameters for practical reasons, despite the fact that the number of continuous dimensions in the belief space is uncountably infinite. Therefore, beliefs must usually be approximated except in special cases, e.g. Kalman filtering may be employed for exact inference in linear Gaussian systems [97]. The task of bounding the error between approximate state-estimates and the Bayesian state-estimate has been the subject of some research (e.g. [24][85]), although thorough investigations on the effects of these approximations on POMDP solutions are scarce due to the complexity of the problem. Therefore, most continuous POMDP solutions operate under the assumption that the state-estimation method provides a reasonable approximation of the Bayesian state-estimate, while it is left to the user to determine whether this assumption is valid. Even under the assumption that an accurate Bayesian state-estimate can efficiently be obtained, planners for continuous POMDPs still need to contend with infinite-dimensional state, action, and observation spaces – which makes the direct application of algorithms designed for discrete POMDPs impractical.

While continuous POMDPs are generally more complex than discrete POMDPs, certain properties of continuous state spaces can sometimes be leveraged to develop efficient solutions. Among the tools that are often exploited in continuous domains are Gaussian probability distributions and the differentiability of continuous functions.

Linear Quadratic Gaussian (LQG; [7]) control on linear Gaussian systems is a prominent example in which an exact solution can be calculated efficiently by applying a linear quadratic regulator to the mean of the belief state that is estimated by a Kalman filter. Through the use of differentiation to linearize nonlinear systems, LQG can be extended to approximate solutions in nonlinear systems as well (e.g. [79][112][117][118]), which is fundamental to the iLQG algorithm presented in Chapter 4. In general, however, exact solutions to general continuous POMDPs – which have nonlinear dynamics, nonlinear measurement functions, and non-Gaussian beliefs – are intractable to find.

2.2.2.1 Point-Based Methods

One intuitive approach to solving continuous POMDPs is to discretize the state space and apply a discrete POMDP method, e.g. [87]. However, this is only possible for small state spaces, since the number of samples that are necessary to sufficiently discretize the state space is subject to the curse of dimensionality [80][128]. Therefore, sampling-based methods which sample directly from the belief space – rather than discretizing the state space – have been proposed to help mitigate the curse of dimensionality. Thrun [110] proposed a Monte Carlo method in which sample beliefs are generated using a particle filter. Then, value iteration is used at the sampled beliefs. When a belief is sampled for which the value function has not been defined, the value is interpolated from the nearest neighbors based on Kullback-Leibler divergence (KL divergence; [49]) if the beliefs are sufficiently similar. If the beliefs of the nearest neighbors are too dissimilar, the sampled belief is added to the set of beliefs at which value iteration is evaluated. This results in a growing set of beliefs over which value

iteration is performed, which becomes unwieldy if the dimensionality of the state space is large or over long planning horizons.

Point-based methods similar to those used in discrete spaces have also been adapted for continuous spaces. For example, the Perseus algorithm [102], which was originally designed for discrete domains, was extended to work in continuous spaces by representing observation, transition, and reward models using Gaussian mixtures; while the beliefs could be represented by Gaussian mixtures or particle sets [80]. The value function could then be parameterized using a set of α-functions that are defined over the state space, which are analogous to α-vectors in discrete domains. Similar to α-vectors, α-functions can be used to define a piecewise linear and convex value function for systems with continuous state spaces, but discrete actions and observations. Efficient sampling-based methods have been proposed for the case in which the action and observation spaces are continuous. The number of α-functions grows exponentially with each value iteration step, which limits the length of the planning horizon.

2.2.2.2 Trajectory Optimization Methods

Since finding the globally optimum solution in POMDPs is generally intractable, trajectory optimization methods instead focus on finding locally optimal solutions by iteratively improving a nominal trajectory. Trajectory optimization methods are initialized with an initial policy that is used to generate an initial nominal trajectory. Since the policy is a function of the belief state and future belief states are unknown until real observations are made, the sequence of future actions and observations is uncertain. Therefore, many methods assume the agent will receive the maximum-likelihood observations, which implicitly results in a nominal trajectory corresponding to maximum-

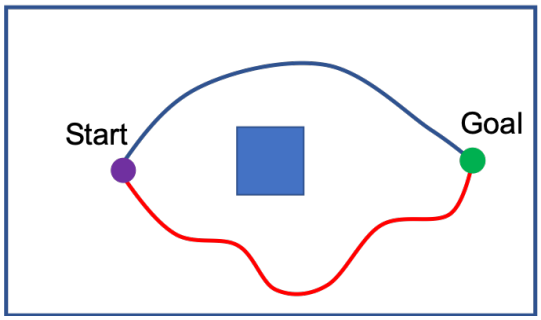
likelihood belief states and deterministic belief state dynamics (e.g. [31][63][78][79][112]. Frequently, state estimation is performed via a Kalman filter (e.g. an extended Kalman filter [121]) under the assumption that the belief state may reasonably be approximated as Gaussian in iterative Linear Quadratic Gaussian (iLQG; [112]) methods, although algorithms which apply particle filters to perform state estimation have also been proposed (e.g. [79]). To account for stochastic belief state dynamics, van den Berg et al. [118][119] extended the iLQG algorithm to account for Gaussian distributed observations rather than assuming maximum-likelihood measurements. The resulting nominal trajectory then represents the means of Gaussian-distributed belief states. The work of van den Berg et al. [118] forms the basis of the algorithm presented in Chapter 4, where modifications to van den Berg's original algorithm are made to improve efficiency. The original algorithm may be found in Appendix A.

Direct optimization methods, such as shooting and collocation methods, have also been applied to the belief space [73]. These methods have been shown to be more effective in dealing with state and control constraints than dynamic programming methods (such as iLQG), although they also rely on the undesirable assumption of maximum-likelihood observations.

2.2.2.3 Sample-Based Path Planners

Sample-based path planners such as PRM [47] and RRT* [46] have also been extended into belief space variants [4][19][84]. Sample-based path planners operate by sampling nodes from the belief space and constructing a graph from which an optimal trajectory can be planned. However, some additional form of a control must be assumed along the edges between nodes. This is often accomplished through the use of a trajectory

Figure 2: Due to the presence of the obstacle, the blue trajectory cannot be continuously deformed into the red trajectory, indicating that they belong to different homotopy classes. In the case of the outfielder problem, obstacles do not exist, so any running path may be continuously deformed into any other running path.



optimization method (see Section 2.2.2.2). However, the existence of a two-point boundary value problem solver that can connect any two sampled configurations may be required, which may be computationally expensive to compute for nonholonomic robots [108].

While trajectory optimization methods continuously deform the trajectory to a locally-optimal trajectory, sample-based path planners advantageously converge to the globally optimal solution (of an approximate system), which makes them particularly useful in systems with many homotopy classes [2]. Thus, sample-based path planners have been applied to problems such as planning collision-free paths (e.g. [3][2][5][66]), determining where to look for optimized autonomous rover localization [70][105], and in simultaneous localization and mapping (SLAM; [30]) to maximize information gain [67][104], to name a few. In this work, sample-based path planners were not considered because the environment is free from obstacles, so any initial trajectory may be continuously deformed into any other feasible trajectory, which makes the use of a trajectory optimization method sufficient[4] (see Figure 2).

[4] Although the methods used in this paper are still susceptible to converging to a locally optimal trajectory rather than the global one, each trajectory may be continuously deformed into any other trajectory and thus belong to the same homotopy class.

2.2.2.4 Other Methods

In simultaneous localization and mapping (SLAM; [30]), an agent is tasked with the problem of mapping an unknown environment while simultaneously localizing itself within the generated map. POMDPs for SLAM problems are complicated by the fact that the environment is unknown, which requires the agent to simultaneously take actions to learn the environment and perform localization while also completing its primary objective. Many SLAM algorithms focus on information gathering (e.g. [104][116]), in which the goal is to map the environment efficiently, although others take on more complicated objectives, such as retrieving a roaming target [36]. In this work, the environment is well defined, so methods which operate in unknown environments are beyond the scope of this work.

2.2.3 HIERARCHICAL METHODS

The human thought process can be used to illustrate how hierarchical methods (e.g. [44][76][113]) can be used to solve POMDPs in both discrete and continuous domains. An example posed by Kaelbling [44] may be paraphrased like this: rather than evaluating 3-dimensional coordinates of a cup, one may simply ask whether or not the cup is in the cupboard. Thus, a set of 3-dimensional coordinates may be abstracted to a logical state "*in the cupboard*." Planning can then be performed based on small set of logical and symbolic variables, rather than in large and complex discrete or continuous domains. However, defining abstract representations and operators for the symbolic states that result in robust behavior is a nontrivial problem which demands more research. Additionally, reliable methods of automating the design of abstract representations do not exist [18], so it is a task that must be completed by humans.

3 OUTFIELDER PROBLEM BACKGROUND

Researchers have posited many different approaches to the outfielder problem that rely on either predictive or heuristic approaches. The goal of each approach is to resolve how a human could intercept a fly ball given their limited resources and limited time in which to make a decision. In addition to the methods presented here, model-free reinforcement learning has also been proposed to resolve the outfielder problem (e.g. [38]). Model-free reinforcement learning methods (including model-free methods which are applied to simulated models) generally require a large number of trials to converge to a good policy, which hinders their application to high-dimensional continuous systems [25][28]. Thus, planning algorithms which have full access to the dynamics generally outperform model-free reinforcement learning methods in these domains [52]. While intuitively it seems likely that humans implement some form of reinforcement learning in the outfielder problem, the specific mechanisms which enables humans to do so are not well understood [51], and so reinforcement learning methods will not be discussed at length in this work.

3.1 TRAJECTORY PREDICTION

The most intuitive approach to the outfielder problem is likely model-based trajectory prediction. In this approach, it is postulated that humans have an internal model of fly ball trajectories and are able to use their vision and other sensory information (e.g. sound of the bat hitting the ball and odometry) to predict the most likely landing spot of the ball and run to it. Such a model would have to accurately predict the ball's position, velocity, and spin as well [88]. Saxberg [89] and Todd [111] showed that humans are poor estimators of landing distances of computer simulated fly balls traveling in parabolic

trajectories, but Saxberg [89] did show that humans demonstrated some predictive abilities. Babler and Dannemiller [8] cautioned that inferring human performance in estimating actual fly ball trajectories based on simulations using 2D displays (e.g. as in [89] and [111]) may be unwarranted. To date, no research has been performed testing humans' abilities to estimate the landing spot of fly balls under realistic conditions, i.e. humans tracking fly balls from the field with their sight of the ball being occluded at various times before impact, and then being tasked with positioning themselves at their prediction of the landing spot of the ball[5].

Shaffer and McBeath [93] have shown that humans generally do not have a very good model of the ball's trajectory. In their experiments, they had both novice and skilled fielders try to identify the time at which the ball reached the apex of its trajectory when viewing actual fly balls from various perspectives. The results indicate that both novice and skilled fielders biased their estimates towards the optical apex rather than the true

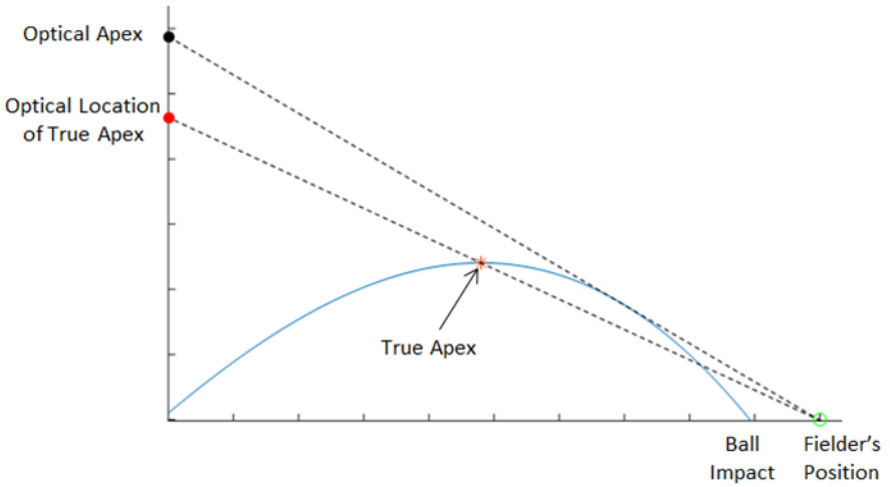


Figure 3: The optical apex, or the highest point when viewed from the perspective of the fielder, occurs later in the trajectory than the time at which the ball physically reaches its highest point in the trajectory.

[5] There are, of course, valid safety concerns which must be addressed before the execution of such an experiment.

apex. Additionally, when the ball was headed directly toward the fielder, so that the optical apex occurred very late in the flight of the ball (if there was one at all), humans seemed to rely on other visual indicators that signified the ball was approaching (e.g. binocular parallax and the size of the image of the ball on the eye) to infer the ball was no longer traveling upward. However, this information is perceivable only shortly before impact, and thus is a poor indicator of when the ball reaches its true apex. This indicates that even if humans do utilize an internal model, then either it is a poor representation of actual physics, or humans cannot accurately estimate the states of a high-fidelity model due to their lack of sensing capabilities or their inability to quickly and accurately propagate and update a state estimate due to time and resource constraints.

Fink et al. [32] had skilled fielders wear a virtual reality headset to catch simulated fly balls with perturbed trajectories. Based on how the fielders reacted to the simulated perturbations, Fink et al. [32] concluded that there is a lack of evidence supporting the hypothesis that humans implement trajectory prediction based solely on initial conditions. The argument presented by [32] was that if fielders estimated the landing spot of the ball based on the ball's initial conditions (as suggested by [88]), then fielders should ignore in-flight perturbations to the ball's trajectory and run to the predicted landing spot of the ball. However, extrapolating this conclusion to infer that humans do not use any trajectory prediction is not justified, as humans may trust new measurements more than previous predictions in the determination of the ball's landing spot.

The most rigorous attempts yet to implement trajectory prediction were performed by Belousov et al. [11] and Höfer [38]. Belousov et al. [11] developed a model that

simulated catching a fly ball traveling in a parabolic trajectory using a covariance-free shooting method which assumed maximum-likelihood observations. The results seemed to demonstrate reasonably human-like behavior, although they have not yet been compared to actual human data. The ball trajectory model used by Belousov et al. [11] neglected aerodynamic effects on the baseball, i.e. drag and Magnus forces. It was also assumed that the agent could directly measure the full global position of the ball with equal noise in each direction (although the noise was state-dependent); while in most camera models the fielder would not be able to measure the ball's depth as reliably as its relative direction.

Additionally, while the fielder was subject to process noise, the fielder was able to fully observe its global position and orientation at each observation, which together provides an unrealistic noise model for a practical fielder. In the work of Höfer [38], a version of iLQG which assumes maximum likelihood observations (as given by [112]) was used as a model predictive controller. In [38], it was also assumed that the fielder could directly measure the ball's global position. Additionally, the orientation of the fielder was assumed to be fixed and the global position and velocity of the fielder was also directly measured (although with noise). Aerodynamic effects were also included, but while the drag force had a precise motion model, the modeling of the Magnus force was relegated to Gaussian noise applied to the ball's trajectory. It should be noted that in this work, parabolic trajectories are assumed, although more realistic motion models should be the subject of future work.

3.2 OPTICAL ACCELERATION CANCELLATION (OAC)

Other researchers have sought methods in which a fielder could move to the correct spot for interception without the need of a full-state representation of the fly ball trajectory. The first such method was proposed by Chapman [21], in which the clever insight was provided showing that if the fielder stood at the landing spot of a ball traveling in parabolic flight, then the tangent of the elevation angle, α, of the ball with respect to the fielder increases at a linear rate until the time of interception (see Figure 4a). An additional observation was made about the case in which the fielder's initial position does not coincide with the landing spot but is in the same plane as the ball's motion: if the fielder moves at the correct constant velocity that will result in interception, then $\tan[\alpha[t]]$ increases linearly until the time of interception, and only the correct constant velocity would cause the $\tan[\alpha[t]]$ to increase linearly (see Figure 4b). Therefore, Chapman proposed that fielders modulate their speed so that $\tan[\alpha[t]]$ increases at a constant rate. This strategy later became known as Optical Acceleration Cancellation (OAC), since $\tan[\alpha[t]]$ increasing at a constant rate can equivalently be interpreted as the ball rising with zero acceleration in the image of a pinhole camera.

$$optical\ acceleration = \frac{d^2 \tan[\alpha[t]]}{dt^2} \tag{7}$$

While Chapman's result was insightful, there are some significant limitations in Chapman's assumptions. First, Chapman assumed the ball travelled in a parabolic trajectory, but aerodynamic effects are significant in determining the trajectory of a baseball. For example, Brancazio [16] showed that drag alone can reduce the flight distance of the ball by up to approximately 40%, and McBeath et al. [58] showed that

Magnus forces can cause the flight of a baseball to deviate significantly from a parabola, such that some trajectories even demonstrate cusps and loops. Despite the parabolic assumption in its derivation, Dienes and McLeod [29] showed that OAC is a viable strategy for catching the ball even if the ball is not traveling in a parabolic trajectory, but this would require that the fielder run at a non-constant velocity.

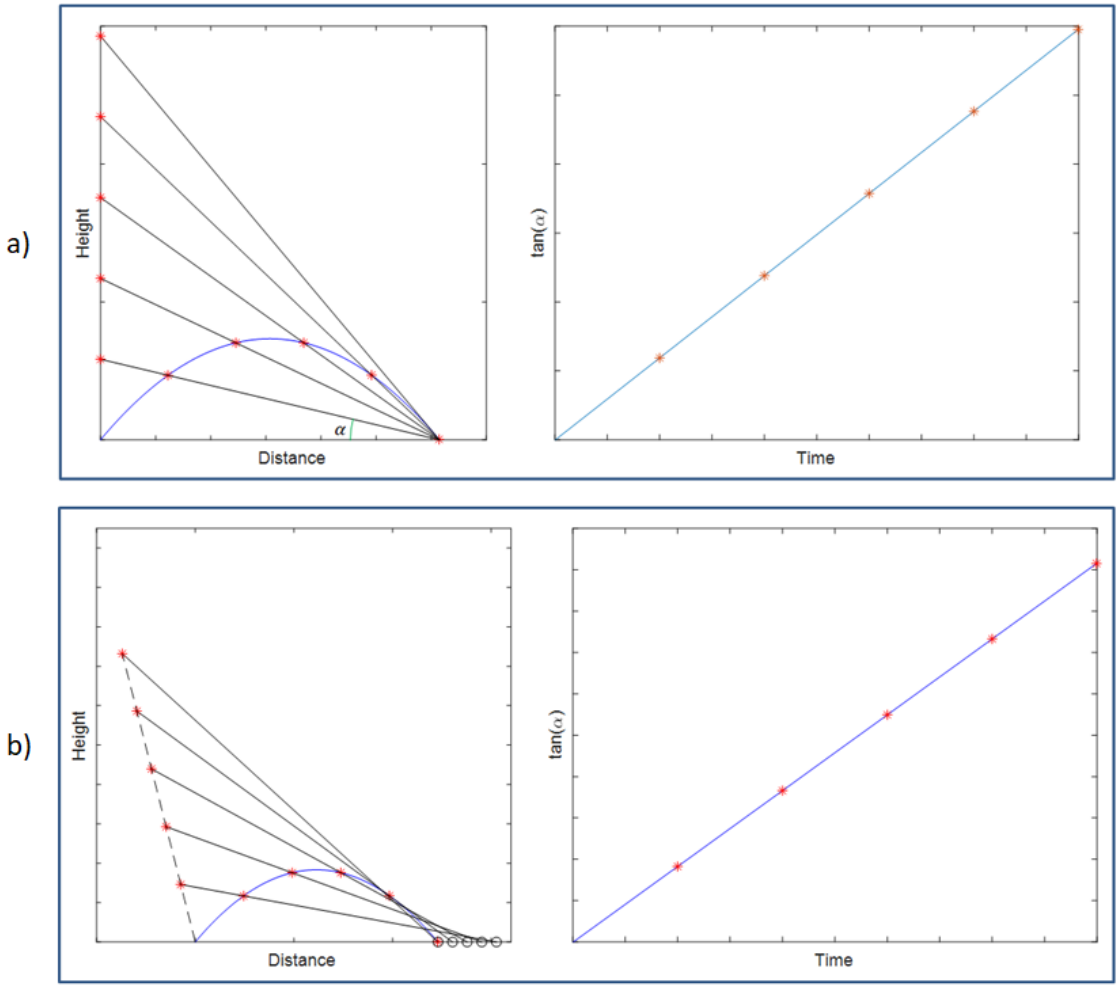


Figure 4: The tangent of the elevation angle, α, increases at a constant rate for a) a stationary fielder at the interception point and b) for a fielder moving at the correct constant velocity to catch the ball. The dashed line slants to the left because the base distance is held fixed at each snapshot in time while the fielder is moving to the left.

The OAC strategy also requires constant feedback in order to determine an appropriate control, thus the fielder cannot lose sight of the ball since the OAC strategy does not incorporate predictions into decision-making. Chapman also admits that it is unlikely that humans perform complex trigonometry (i.e. calculating the tangent of an angle) in their heads while they are running to catch the ball. However, Chapman points out that this heuristic strategy was not intended to describe actual human decision making, but that it was meant to illustrate how humans could exploit the underlying laws of motion to devise a strategy that needs only a small amount of information and computation to be successful.

Chapman's method formed a basis that compelled much further study into the outfielder problem. Todd [111] demonstrated that humans are not particularly sensitive to image acceleration, calling into question whether OAC is a viable interception strategy. However, Babler and Dannemiller [8] showed that humans' sensitivity to optical acceleration is proportional to the optical velocities, and humans can use this information to detect whether a ball will land in front or behind them if a sensitivity threshold was met. Additionally, Michaels and Oudejans [62] and Dienes and McLeod [29] collected empirical data of fielders catching actual fly balls, and observed that the fielders' running paths nulled the optical acceleration of the ball, as predicted by OAC. It remained uncertain though if this was being done deliberately by the fielder, or if it was just a byproduct of a different strategy being performed by the fielder.

The OAC strategy only has the capability to account for the fielder's behavior if the fielder's initial position is in the same plane as the ball's trajectory. For the more typical scenario in which the ball will land to the side of the fielder, additional

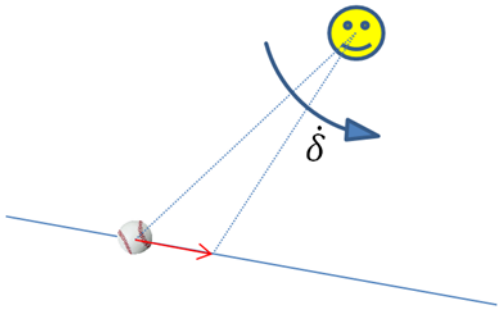


Figure 5: Top-down view of the fielder and the ball's trajectory. The angular rate $\dot{\delta}$ is the rate in which the fielder must rotate to fixate on the ball as used by Tresilian (1995) and McLeod et al. (2006).

considerations must be employed for controlling lateral movement. Chapman [21] suggests that the fielder maintains OAC for forward and backwards motion, and a constant bearing angle to control lateral motion, although no empirical evidence was provided supporting its implementation by humans.

Tresilian [114] interpreted a constant bearing angle as implying that the angular rate of the bearing angle $\dot{\delta}$ is zero. Tresilian's method therefore utilized the angular rate $\dot{\delta}$ in which the fielder had to rotate in order to fixate on the ball in conjunction with OAC. In this method, the fielder would use OAC to calculate the desired acceleration of the fielder in forward and backward motion. Then, a desired acceleration that is proportional to the rate in which the fielder must rotate to fixate on the ball is applied in the direction perpendicular to the one calculated by OAC. The desired acceleration of the fielder is then determined from the sum of the two accelerations (with some additional considerations to ensure certain ad hoc thresholds are not exceeded). This method caused the simulated fielder to approach the plane of the ball's motion faster than Chapman's method, although no empirical evidence was provided to suggest that this method was implemented by humans.

Jacobs et al. [41] support the view that fielders run to the plane of the ball's motion first, and then use OAC to adjust their position within the plane of the ball's motion. However, they did not provide a method as to how this could be accomplished and provided supporting data from only one trial. The work of Tresilian [114] was generalized by McLeod et al. [61] to form the Generalized Optical Acceleration Cancellation (GOAC). In the GOAC strategy, the fielder uses OAC to control $\tan[\alpha[t]]$. The fielder then varies their lateral movement based on the angular rate $\dot{\delta}$ that they have to turn left or right to face the ball, similar to Tresilian [114]. The GOAC strategy states that $\dot{\delta}$ is controlled to be a constant value when the fielder is close to catching the ball. Thus, the fielder seeks to null the angular acceleration required to face the ball, rather than nulling the angular velocity as was done by Tresilian [114]. Thus, the direction that the fielder attempts to run when using GOAC is the direction in which $\tan[\alpha[t]]$ and $\dot{\delta}$ both increase linearly. Additionally, McLeod et al. [61] provided some empirical evidence that supports the use of GOAC by humans. However, the GOAC strategy does not describe precisely how a fielder should behave early in the ball's trajectory, which makes it difficult to implement as a control strategy.

3.3 LINEAR OPTICAL TRAJECTORY (LOT)

McBeath et al. [56] noted that while the research of Todd [111] and Babler and Dannemiller [8] demonstrated that generally humans are poor at detecting optical acceleration, humans have demonstrated better proficiency at detecting optical curvature, i.e. whether a line is straight or curved. McBeath et al. [56] used this principle to motivate the Linear Optical Trajectory (LOT) heuristic. Similar to the OAC heuristic, the LOT heuristic requires no knowledge of the distance to the ball or home plate. The intent of

the LOT heuristic is for the fielder to run along a path such that the trajectory of the ball forms a straight line within the fielder's image plane, due to humans having demonstrated proficiency in detecting optical curvature.

Let ψ be the angle between the ground plane and the line from home plate to the ball as projected onto the fielder's initial image plane (see Figure 6a.). Provided that the horizon is always oriented in the same direction in the image, the angle ψ can be described using the image coordinates of the ball (u_b, v_b) and home plate (u_h, v_h):

$$\psi = \tan^{-1}\left[\frac{u_b - u_h}{v_b - v_h}\right] \tag{8}$$

While it is possible to generalize the definition of ψ into the case in which the image plane rotates in a manner that changes the orientation of the horizon, it is not necessary for the fielder model considered in this work (see Section 5.2).

The LOT heuristic specifies that ψ must remain a constant value. In McBeath et al. [56], the angle ψ in the LOT heuristic is geometrically related to the angle of elevation, α, as employed by OAC strategy, and a lateral angle β, which may be described as the horizontal angle between the direction from the fielder to home plate and the direction from the fielder to the vertical projection of the ball onto the ground. This relation is given by the following equation.

$$\tan[\psi] = \frac{\tan[\alpha]}{\tan[\beta]} \tag{9}$$

The geometric reasoning for this relation is illustrated in Figure 6a. using a trirectangular tetrahedron, i.e. a tetrahedron in which all three face angles at one vertex are right angles. Equation 9 may be extended to be time-varying values.

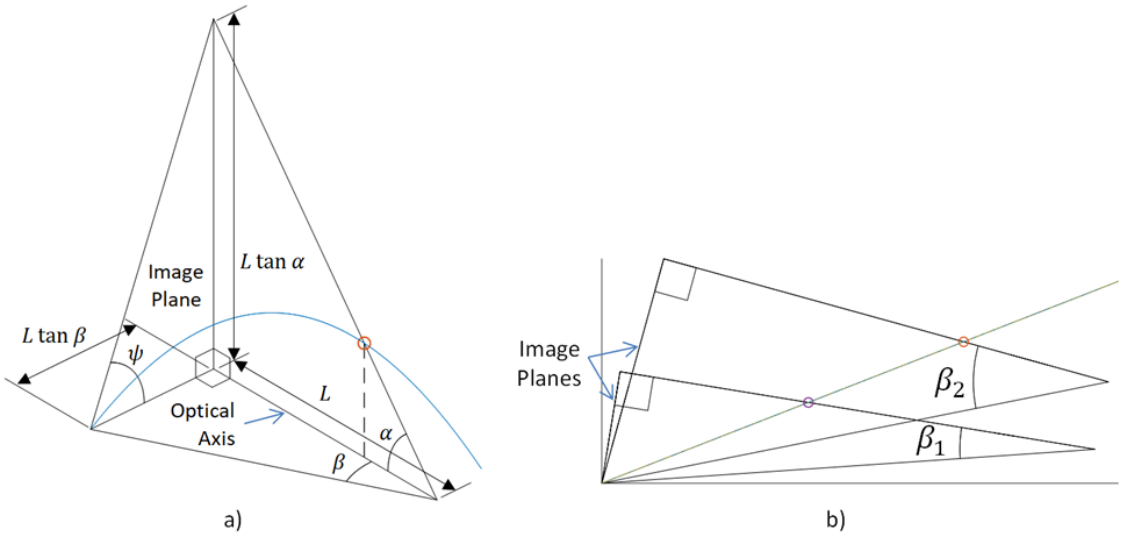


Figure 6: a) Geometric relationships used in the LOT heuristic represented using a trirectangular tetrahedron. The angle ψ in the image plane is controlled to be constant even as b) the fielder rotates to fixate on the ball.

$$\tan[\psi[t]] = \frac{\tan[\alpha[t]]}{\tan[\beta[t]]} = \frac{f[t]C_\alpha}{f[t]C_\beta} = C_\psi \tag{10}$$

In the following section, it will be shown that this implies a specific orientation of the image plane in which the fielder's gaze tracks the ball laterally, but not vertically.

The LOT heuristic that McBeath et al. [56] propose additionally requires $f[t]$ of Equation 10 to increase monotonically. Thus, if the $\tan[\alpha[t]]$ increases monotonically and proportional to $\tan[\beta[t]]$, then it is equivalent to the ball traveling along a monotonically increasing optical trajectory. Since an infinite number of functions satisfy the monotonic constraint, there are infinite running paths that satisfy the LOT heuristic. McBeath et al. [56] suggest setting $f[t]$ to be a linear function for ball trajectories which are approximately parabolic. Setting $f[t]$ to be a linear function is equivalent to using the OAC heuristic to control the elevation angle, and the LOT heuristic to control lateral motion of the fielder – although $f[t]$ is not constrained to be linear for the

implementation of LOT in general. McBeath et al. [57] concede that the LOT heuristic alone is not sufficient to lead to interception, as the function $f[t]$ must be chosen appropriately.

Dannermiller et al. [26] contested that much of the research cited by McBeath et al. [56] in demonstrating that humans are good at detecting curvature were for lines in which the human subject was presented with full view of the line at once, rather than a streaking point moving along a line. Thus, the LOT strategy is more of a spatiotemporal heuristic rather than a strictly spatial one, as assumed by the supporting evidence presented by McBeath et al. [56]. While this does not necessarily imply that humans would be ineffective at achieving LOT, the evidence presented in McBeath et al. [56] is not sufficient to demonstrate that humans would be effective in such a task and thus further study is needed. Jacobs et al. [41] observed that near the end of the ball's flight, fielders often arrive at the landing site of the ball or align themselves with the plane of the ball's travel and move slightly radially in order to make the catch, neither of which can be accounted for using LOT theory. McBeath et al. [57] note that the LOT model was only intended for use in the initial part of the ball's flight and not during the final descent. During the final descent of the ball, humans may use other cues, such as optical enlargement and stereo disparity, which the LOT heuristic does not take into account.

McLeod et al. [59] show that for some fielder trajectories, $\tan[\alpha[t]]$ increases linearly as predicted by OAC and LOT. However, $\tan[\beta[t]]$ increased linearly for some catches but not for others, depending on the fielder. Thus, $\tan[\beta[t]]$ did not increase proportionally to the linearly increasing $\tan[\alpha[t]]$, which is inconsistent with the predictions of the LOT heuristic. Additionally, McLeod et al. [60] note that some data

provided by Shaffer and McBeath [90] seems to indicate that fielder motions are counter to the predictions of LOT for fly balls that are uncatchable. Shaffer et al. [91] clarify that the LOT heuristic is meant to break down for uncatchable fly balls, and the manner in which fielders behave when the LOT heuristic fails is one in which the ball lands in front of the fielder rather than behind.

The LOT heuristic is also not specific about what control is used to correct optical curvature. McBeath et al. [56] suggest that humans react to correct observed upward or downward curvature of the optical trajectory. However, it is not clear if curvature is meant in the sense of differential geometry [106], or if another quantity which may indicate a deviation from a straight line (e.g. orthogonal distance to the desired linear optical trajectory) is controlled as a means to correct curvature. It may also be reasonable deduced that the fielder is intended to directly control ψ to be a constant value. In the following section, a novel formulation of a control variable which results in the satisfaction of the LOT heuristic is proposed.

3.4 GENERALIZED LOT

The intent of the LOT heuristic presented by McBeath et al. [56] is for the fielder to choose a running path such that the ball forms a linear optical trajectory in the image plane. However, the geometric relation described in Equation 9 and illustrated in Figure 6 necessitates that a linear optical trajectory will always be observed, independent of the path of the fielder. This is due to the manner in which the fielder rotates the image plane to track the ball. Referring to Figure 6, which is similar to one that is presented in McBeath et al. [56], it can be seen that the orientation of the image plane is always orthogonal to the ground plane. This implies that the optical axis is parallel to the ground

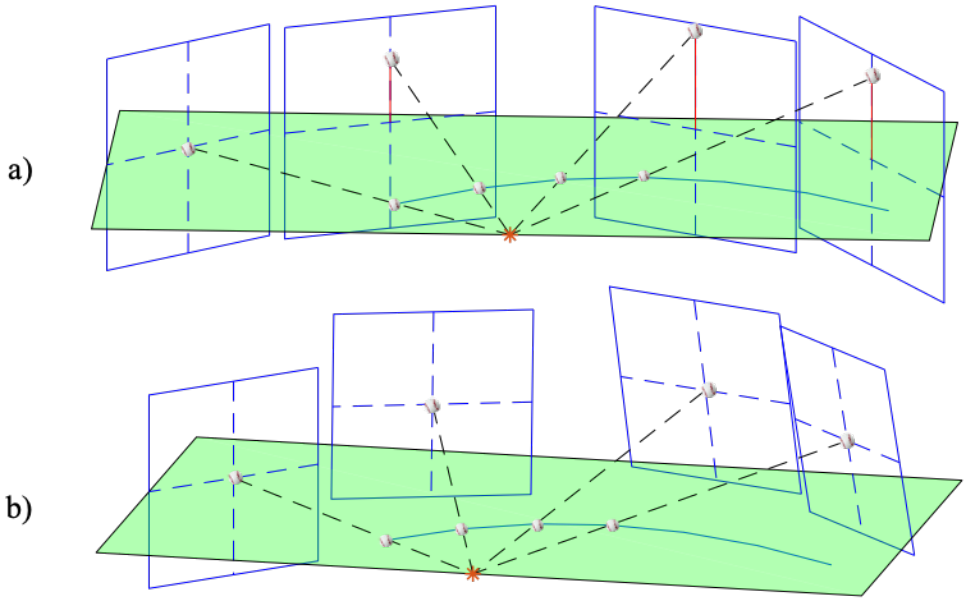


Figure 7: Depending on the fielder's rotation, the fielder may necessarily observe a linear optical trajectory (although not necessarily monotonically increasing). In a), the geometric relation provided by Equation 9 implies that the fielder rotates such that the ball is constrained to strictly vertical motion in the fielder's image plane. In b), the fielder's gaze is fixated on the ball, so that the ball is always at the center of the image plane.

plane since the optical axis is orthogonal to the image plane. The fielder rotates so that the optical axis remains vertically aligned with the ball such that the ball stays at the horizontal center of the image plane, which may be inferred from the right angle in the image plane in Figure 6. Since the position of the ball is constrained to the horizontal center of the image plane, this method would result in a ball which rises and falls vertically in the image plane. Thus, the ball must necessarily follow a linear optical trajectory independent of the translational motion of the fielder. A later paper [120] describes the geometry in a slightly different manner, in which the image plane rotates such that the ball is aligned with the optical axis, i.e. the gaze of the fielder is fixed on the ball. This would imply that the ball would always remain fixed at the origin of the image

plane – independent of the path of the fielder – rather than forming a linear optical trajectory. Similar to McBeath's original paper [56], the heuristic demands that the fielder chooses a running path to maintain constant ψ. While the optical trajectory of the ball in each of these scenarios is trivialized by fielder's rotation of the image plane, the angle ψ is also dependent on the direction to home plate – which adds additional degrees of freedom that need to be controlled. Thus, maintaining constant ψ requires translational movement of the fielder for any choice of rotational motion of the image plane. The constraint that ψ is constant describes a family of possible LOT heuristics, with the running path in each being determined by how the fielder chooses to rotate the image plane.

The constraint that ψ is constant implies its derivative with respect to time $\dot{\psi} = 0$. Referencing Equation 8, this implies

$$\dot{\psi} = \left(\dot{r_u} \frac{r_v}{r_u^2 + r_v^2} - \dot{r_v} \frac{r_u}{r_u^2 + r_v^2} \right) = 0 \tag{11}$$

where $r_u = u_b - u_h$ and $r_v = v_b - v_h$ are the relative distance components between the ball and home plate in image coordinates. With some algebraic manipulation, it can be seen that

$$\frac{r_u}{r_v} = \frac{\dot{r_u}}{\dot{r_v}} \tag{12}$$

which implies

$$\boldsymbol{r} \propto \dot{\boldsymbol{r}}, \qquad \text{where } \boldsymbol{r} = \begin{bmatrix} r_u \\ r_v \end{bmatrix} \tag{13}$$

Therefore, for the LOT heuristic to be satisfied, the relative image velocity between the ball and home plate must be in the same direction as the relative image displacement

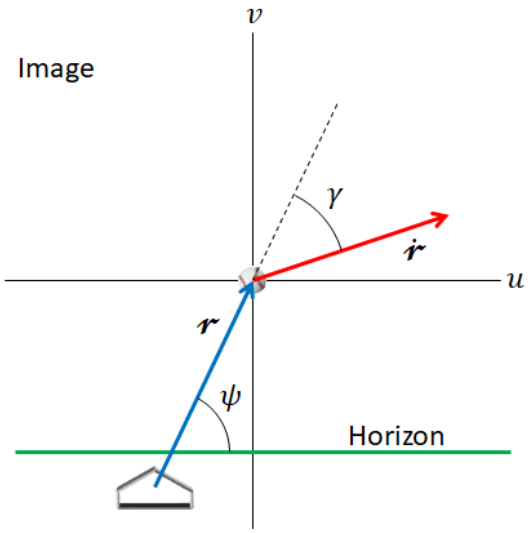


Figure 8: The Generalized LOT heuristic requires that the relative image velocity, $\dot{\boldsymbol{r}}$, between the ball and home plate to be in the same direction as the relative image displacement $\boldsymbol{r}$ (i.e. $\gamma = 0$). Satisfaction of this constraint would cause ψ to be a constant value.

within the fielder's rotating and translating image plane. By letting γ be the angle between $\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$, the *generalized LOT heuristic* that is proposed here is $\gamma = 0$, which would imply $\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$ are in the same direction.

From a practical standpoint, it does not seem efficient for the fielder to keep track of the directions of both the ball and home plate. Results from [120] seem to indicate that background movements affect the running paths of human fielders, thus it is possible that humans utilize background information to help satisfy the LOT heuristic. In this work, it is assumed that the background is featureless, so that it does not provide any additional information to the fielder. The fielder also cannot reference home plate as a navigational aid. Therefore, the fielder must have a sense about their own global state[6] in order for the fielder to know the direction to home plate and satisfy the generalized LOT heuristic. This is in contrast to Tresilian's method [114] (see Section 3.2), which does not require

[6] In this work, it is assumed that the fielder has access to a full Bayesian state estimate of their global position; see the discussion at the beginning of Chapter **Error! Reference source not found.** for further explanation.

the fielder to maintain global information but rather only local information about their current angular rate.

3.5 GENERAL DISCUSSION ABOUT CATCHING HEURISTICS

The aforementioned catching heuristics describe the fielder behavior that will be observed in successful catching strategies, but do not elaborate much on the type of controller that would cause the heuristic to be satisfied. Generally, proportional or proportional-derivative control seems to be the implied controller ([21][32][38][56][61]), although the methodology to select the proper gains is not clear. The McRuer-Krendel controller has also been proposed for modeling to account for human reaction time [114].

There have been several studies that have evaluated the feasibility of the various fielding strategies from an analytic viewpoint, yet several assumptions remain in previous work that are still too restrictive to assess whether they would be successful in real-life, either for human or mobile robot implementations. Previous work has primarily evaluated the performance of heuristics on either parabolic trajectories (e.g. [11][21][56][59]) or trajectories with drag (e.g. [38][114]). However, Magnus forces significantly affect the flight of a baseball [58]. For example, cusps or even loops can be introduced to the trajectories of fly balls that are hit high but do not travel far [58]. Additionally, Magnus forces can also cause significant lateral curvature. No analysis of the outfielder problem to date has adequately modeled these effects which are introduced by the Magnus force in 3-dimensions, although McBeath et al. [58] have observed that the OAC heuristic sometimes requires sudden fielder movements in the 2-dimensional case. This work assumes a parabolic trajectory, with emphasis on improving the measurement and noise models instead of modeling the ball's trajectory with greater fidelity. However, modeling

of the ball's trajectory with greater fidelity is an important consideration for evaluating the feasibility of catching heuristics in future work.

Previous research on fielding heuristics has also mostly only covered deterministic systems, while only a handful of studies have evaluated the success of fielding heuristics under random perturbations (e.g. [38][114]). For example, Tresilian [114] applied Gaussian noise to optical acceleration measurements in the 1-dimensional case of OAC being implemented with a McRuer-Krendel controller, to which the heuristic control demonstrated robustness, but results for the 2-dimensional case were not provided. Höfer [38] provided a more thorough analysis by testing the performance of several heuristics for 2-dimensional motion when perturbations were applied to the measurements, the motor control, and the ball's trajectory. Generally, it was found that the performance of common heuristic methods degraded more than predictive methods in the presence of noise, although predictive models were more sensitive to the inaccurate modeling of drag. However, there were a few ways in which the analysis could be improved. First, the measurement model assumed that the global positions of the fielder and the ball could be measured directly with additive noise, while a practical measurement would only provide the relative direction from the fielder to the ball in the fielder's coordinate system. Next, each of the heuristics studied requires that the fielder has some knowledge of their rotation rate, which will be uncertain for any practical fielder – yet this noise was not included in the motion model of the fielder. Finally, all noise terms were varied proportionately, so the effects of individual perturbations could not be discerned. The system model in this work makes improvements in each of these areas.

There has also been debate as to whether the satisfaction of a given heuristic arises merely as a geometric consequence of the implementation of another control strategy [61]. Belousov et al.'s [11] research indicates that the satisfaction of each heuristic may be viewed as geometric consequences of the fielder running along an approximately optimal trajectory that may be determined by their stochastic optimal controller, which is based on the trajectory optimization method presented in [73]. Additionally, Belousov et al.'s [11] stochastic optimal controller accounts for some observed human behaviors which the heuristic approaches that were considered in this work do not: humans tend to exhibit some predictive behavior when catching fly balls (i.e. deliberately taking their eyes off the ball to gain a speed advantage) which cannot be explained using heuristic approaches, while a predictive controller is capable of describing such behavior. However, humans seem to be poor estimators of the ball's trajectory (see Section 3.1) and the stochastic optimal controller is computationally expensive, therefore it is unlikely that humans would be able to implement it. Therefore, Belousov et al. [11] suggest that humans may use various heuristics at the appropriate times to compose an approximately optimal policy.

3.6 CONSIDERATIONS IN GENERAL PRACTICAL DECISION MAKING

Belousov et al.'s [11] hypothesis that a human fielder's control policy is composed of several different heuristics is consistent with the view of Gigerenzer and Selten [35], who suggest that human decision making is performed through the implementation of an "adaptive toolbox" of heuristics. In the adaptive toolbox approach, humans develop quick and easy solutions to hard problems or components of hard problems, and then exploit the heuristics in the toolbox to generate a good solution. The

work of Tversky and Kahneman [115] provided empirical evidence of the use of heuristics in many specific human decision-making situations. They also went on to show that the heuristics implemented by humans often lead to predictable and systematic cognitive biases. Kahneman [45] suggests that the heuristic techniques that are implemented by humans are crucial for timely decision making, and the biases introduced through the use of a heuristic can be mitigated through additional conscious effort if necessary.

The use of heuristics for describing human decision making was first popularized by Herbert Simon [98], who observed that humans are not capable of the classical view of "rational" (i.e. optimal) decision making due to the fact that determining the optimal decision is generally a computationally intractable problem. Thus, optimal decisions seldom can be found practically because humans have only limited memory and computational resources while only having a short time in which to make a decision. Additionally, decision makers seldom have access to complete information (i.e. a proper prior), so the optimization problem is often ill-defined. Therefore, Simon argued that practical decision makers (e.g. humans or artificial intelligence) can only hope to achieve *bounded rationality*. Bounded rationality is an idea developed by Simon that implies that the rationality of a practical decision maker is limited by the intractability of the problem, the computational power of the decision maker, the time available to make a decision, and the information available to the decision maker. Simon suggests that humans cope with their bounded rationality through *satisficing*, a cognitive heuristic in which the decision maker searches for a satisfactory solution rather than an optimal one. The

threshold of what is considered to be satisfactory can also recede as time progresses, depending on the problem, to ensure a timely decision is always reached.

It has also been hypothesized that humans reduce the amount of computation that they have to perform through simplifying the state representation that is used to make a decision, which is sometimes called the *controlled variable* in cognitive science [55], and is analogous to the *agent state* in reinforcement learning [109]. In the limit, the state representation can be reduced to only that which can be directly measured or perceived, which has been called *perceptual control* by some psychologists [81]. For example, it has been observed that honeybees perform direct feedback on optical flow to regulate forward and vertical velocity during landing without explicitly estimating either state [103]. This concept is also applied by roboticists in many image-based visual servoing techniques [23]. Image-based visual servoing allows a robot's end effector to be precisely positioned in 3-dimensional coordinates without any *a priori* knowledge of the 3-dimensional coordinates of objects in its workspace. Additionally, the error-correcting feedback provided by visual servoing allows good performance without a high degree of mechanical accuracy of the servos, which demonstrates how a well-designed state representation can also provide robust performance.

Humans also have demonstrated the ability to leverage the environment to assist in computation, thereby enabling faster decision making through the use of *epistemic actions*. Kirsh and Maglio [48] define epistemic actions to be "physical actions that make mental computation easier, faster, or more reliable." This is in contrast to what they define as *pragmatic actions*, which are actions that directly move the agent closer to a desired goal. In a prominent experiment, subjects were observed while playing Tetris to

determine if humans performed actions that only had epistemic value. An epistemic action could be clearly distinguished from a pragmatic action if the action that was performed was deliberate (not a mistake), did not bring the agent closer to the goal state, and a reasonable explanation could be given that described how the action simplifies the computation required by the human. For example, it was noticed that even expert Tetris players made superfluous rotations, leading the researchers to infer that a physical rotation of a Tetris piece could be performed much faster and more reliably than a mental rotation to compute matching contours. While the Tetris experiment demonstrated that some human actions can be best described as having only epistemic value, in many applications a single action can serve both epistemic and pragmatic functions.

Epistemic actions are more generally cast as an example of *embodied cognition* in cognitive psychology, in which the environment is used to aid in the cognitive process. It has been proposed that humans employ embodied cognition in the outfielder problem to choose running paths that minimize their cognitive load [124], and such decision-making would thus be epistemic in nature. For example, choosing a running path in which the optical acceleration is always zero eliminates the need for the fielder to calculate where the ball will land. Unfortunately, these hypotheses are not tested in this work, as the theory of epistemic actions demands more rigorous mathematical development before this is possible. Further research may determine that epistemic actions are composed of already familiar concepts (e.g. such as belief compression), or it may be determined that a new theory of epistemic actions allows the development of new innovative ways for efficient decision making, but that is not decided in this work.

Various researchers have previously attempted to include resource allocation (e.g. computation time and memory) in the objective functions for optimization (e.g. [13][39]). However, this often leads to optimization problems with even greater complexity than the original problems they were meant to simplify, which makes practical solutions impossible to find. A new theory of epistemic actions may permit new approaches to approximate solutions to such problems.

4 MODIFIED BELIEF ILQG

In this work, van den Berg et al.'s belief space variant of iLQG [118] was implemented to find an approximately optimal solution to the outfielder problem due to its efficiency in which locally-optimal solutions to the outfielder problem could be found, while also improving on similar approaches which assume maximum likelihood observations (e.g. [11][38]). The original form of the algorithm may be found in Appendix A. It has been modified using directional derivatives to calculate several matrix derivatives required by the original algorithm, which improves the efficiency of the algorithm while performing the same calculation. The policies generated by both forms of the algorithm are equivalent down to the numerical precision of the computer.

4.1 PROBLEM DEFINITION

It is assumed that the state, action, and observation spaces are all continuous, and that that the belief state, process, and measurement noises may be characterized by Gaussian distributions. Since a Gaussian distribution may be parameterized by its mean and variance, let $\boldsymbol{b}[\hat{\boldsymbol{x}}, P] \sim \mathcal{N}[\hat{\boldsymbol{x}}, P]$ indicate a Gaussian belief state parameterized by its mean $\hat{\boldsymbol{x}}$ and variance P. Since the belief $\boldsymbol{b}[\hat{\boldsymbol{x}}, P]$ is fully parameterized by its mean and

variance, it is also sufficient to refer to a belief as the pair $\hat{\boldsymbol{x}}, P$, and any function of this pair of parameters is thus implicitly a function of the belief state. It is also assumed that the system may be described by the following state dynamics and observation model for the random variable $\boldsymbol{x}_k$:

$$\begin{gathered} \boldsymbol{x}_{k+1} = \boldsymbol{f}[\boldsymbol{x}_k, \boldsymbol{u}_k] + \boldsymbol{m}_k, \qquad \boldsymbol{m}_k \sim \mathcal{N}\left[\boldsymbol{0}, M[\boldsymbol{x}_k, \boldsymbol{u}_k]\right] \\ \boldsymbol{y}_k = \boldsymbol{h}[\boldsymbol{x}_k] + \boldsymbol{n}_k, \qquad \boldsymbol{n}_k \sim \mathcal{N}\left[\boldsymbol{0}, N[\boldsymbol{x}_k]\right] \\ \boldsymbol{x}_0 \sim \mathcal{N}[\hat{\boldsymbol{x}}_0, P_0] \end{gathered} \tag{14}$$

where $\boldsymbol{m}_k$ and $\boldsymbol{n}_k$ are the process and measurement noises, respectively, and the initial belief $\hat{\boldsymbol{x}}_0, P_0$ is given. It is assumed that $\boldsymbol{m}_k$ and $\boldsymbol{n}_k$ are independent zero-mean Gaussian distributions which may have state and action dependent variance. The goal is to determine a locally optimal policy $\boldsymbol{u}_k = \boldsymbol{\pi}_k^*[\hat{\boldsymbol{x}}_k, P_k]$ that minimizes the value function:

$$v_0[\hat{\boldsymbol{x}}_0, P_0] = \mathrm{E}\left[c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell] + \sum_{k=0}^{\ell-1} c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]\right] \tag{15}$$

where ℓ is the length of the planning horizon, $c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$ is the cost of the final belief state, and $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ is the immediate cost of executing action $\boldsymbol{u}_k$ at belief state $\hat{\boldsymbol{x}}_k, P_k$. Given that $c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$ and $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ are cost functions, the value function $v_0[\hat{\boldsymbol{x}}_0, P_0]$ would thus represent a cost function. It is required that the Hessians of the cost functions obey the following constraints:

$$\frac{\partial^2 c_\ell}{\partial \hat{\boldsymbol{x}}_\ell \partial \hat{\boldsymbol{x}}_\ell} \geq 0, \qquad \frac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \boldsymbol{u}_k} > 0, \qquad \begin{bmatrix} \dfrac{\partial^2 c_k}{\partial \hat{\boldsymbol{x}}_k \partial \hat{\boldsymbol{x}}_k} & \dfrac{\partial^2 c_k}{\partial \hat{\boldsymbol{x}}_k \partial \boldsymbol{u}_k} \\ \dfrac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \hat{\boldsymbol{x}}_k} & \dfrac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \boldsymbol{u}_k} \end{bmatrix} \geq 0 \tag{16}$$

where $A > 0$ implies A is positive definite, and $A \geq 0$ implies A is positive semi-definite. The constraints in Equation 16 imply that the value function must be a convex function

with respect to the mean of the belief state at all time steps k, and must be a strictly convex function with respect to the input. The requirement that the value function is strictly convex function with respect to the input causes the optimal action to be unique.

4.2 EXTENDED KALMAN FILTER

Since exact Bayesian inference is generally intractable, the Extended Kalman Filter (EKF; [97][121]) is used as the state estimator. The EKF relies on the following first-order approximations for the mean and variance of a function $\boldsymbol{g}[\boldsymbol{z}]$ of a stochastic variable $\boldsymbol{z}$.

$$\mathrm{E}\big[\boldsymbol{g}[\boldsymbol{z}]\big] \approx \boldsymbol{g}\big[\mathrm{E}[\boldsymbol{z}]\big] \tag{17}$$

$$\mathrm{Var}\big[\boldsymbol{g}[\boldsymbol{z}]\big] \approx \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}}\big[\mathrm{E}[\boldsymbol{z}]\big]\mathrm{Var}[\boldsymbol{z}]\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}}\big[\mathrm{E}[\boldsymbol{z}]\big]^T \tag{18}$$

The EKF assumes the initial belief $\widehat{\boldsymbol{x}}_0, P_0$ accurately describes the initial distribution of the random variable $\boldsymbol{x}_0$. Given a belief $\widehat{\boldsymbol{x}}_k, P_k$, the belief at the next time step may be determined as:

$$\widehat{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + K_k\big(\boldsymbol{y}_{k+1} - \boldsymbol{h}\big[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\big]\big) \tag{19}$$

$$\begin{aligned} P_{k+1} &= (I - K_k H_k)\Gamma_k \\ &= (\Gamma_k^{-1} + H_k^T N_k^{-1} H_k)^{-1} \end{aligned} \tag{20}$$

where,

$$\Gamma_k = F_k P_k F_k^T + M_k \tag{21}$$

$$\begin{aligned} K_k &= \Gamma_k H_k^T (H_k \Gamma_{\mathrm{t}} H_k^T + N_k)^{-1} \\ &= P_{k+1} H_k^T N_k^{-1} \end{aligned} \tag{22}$$

$$F_k = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k], \quad H_k = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\big[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\big] \tag{23}$$

$$M_k = M[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k], \qquad N_k = N\big[\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\big] \tag{24}$$

Two forms of the variance update P_{k+1} and the Kalman gain K_k are listed in Equations 20 and 22, respectively. The first form is conventional for Kalman filter form, while the latter is typically used in information filters [97]. Both forms are listed because each will be convenient at different times throughout this work. The latter form is slightly more computationally expensive than the standard form for use in Kalman filtering. However, this form is used frequently in this work because it simplifies the presentation of many analytic partial derivatives. Future work may seek to further improve the efficiency of the algorithm through considering the ordering of matrix operations.

4.3 BELIEF DYNAMICS

Since the measurement that the agent will receive is uncertain, the resultant belief state of an action and observation sequence will be uncertain, as was mentioned in Section 2.2. This may also be deduced from the belief update given by Equations 19-24. First, let $\boldsymbol{w}_k$ be

$$\boldsymbol{w}_k = K_k\big(\boldsymbol{y}_{k+1} - \boldsymbol{h}\big[\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\big]\big) \tag{25}$$

Then, Equation 19 may be rewritten as

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w}_k \tag{26}$$

It is assumed that the mean of the belief state $\hat{\boldsymbol{x}}_k$ at time k is given deterministic variable, and that $\boldsymbol{u}_k$ represents a deterministic input (any stochastic part of the input may be lumped into $\boldsymbol{m}_k$), therefore the first term of the Equation 26 is deterministic. However, $\boldsymbol{w}_k$ is a stochastic variable since the measurement $\boldsymbol{y}_{k+1}$ is uncertain. The first order approximation of the mean of $\boldsymbol{w}_k$ yields

$$\mathrm{E}[\boldsymbol{w}_k] = K_k \mathrm{E}[\boldsymbol{y}_{k+1}] - K_k \mathrm{E}\left[\boldsymbol{h}\left[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\right]\right] \tag{27}$$

where

$$\begin{aligned} \mathrm{E}[\boldsymbol{y}_{k+1}] &= \mathrm{E}[\boldsymbol{h}[\boldsymbol{f}[\boldsymbol{x}_k, \boldsymbol{u}_k] + \boldsymbol{m}_k] + \boldsymbol{n}_k] \\ &\approx \boldsymbol{h}\left[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\right] \end{aligned} \tag{28}$$

which is due to the fact $\boldsymbol{m}_k$ and $\boldsymbol{n}_k$ are zero mean, Equations 14 and 17-18, and since $\widehat{\boldsymbol{x}}_k$ is the mean of $\boldsymbol{x}_k$. Therefore, the $\mathrm{E}[\boldsymbol{w}_k] \approx \boldsymbol{0}$ to first order. The variance is

$$\mathrm{Var}[\boldsymbol{w}_k] = K_k \mathrm{Var}\left[\boldsymbol{y}_{k+1} - \boldsymbol{h}\left[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\right]\right] K_k^T \tag{29}$$

where

$$\begin{aligned} \mathrm{Var}\left[\boldsymbol{y}_{k+1} - \boldsymbol{h}\left[\boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k]\right]\right] &= \mathrm{Var}[\boldsymbol{y}_{k+1}] \\ &= \mathrm{Var}[\boldsymbol{h}[\boldsymbol{f}[\boldsymbol{x}_k, \boldsymbol{u}_k] + \boldsymbol{m}_k] + \boldsymbol{n}_k] \\ &\approx H_k \Gamma_k H_k^T + N_k \end{aligned} \tag{30}$$

since $\Gamma_k = \mathrm{Var}[\boldsymbol{f}[\boldsymbol{x}_k, \boldsymbol{u}_k] + \boldsymbol{m}_k]$. By combining Equations 29-30 and by substitution of the first definition of K_k from Equation 22, it can be seen that

$$\begin{aligned} \mathrm{Var}[\boldsymbol{w}_k] &= K_k (H_k \Gamma_k H_k^T + N_k)((H_k \Gamma_{\mathrm{t}} H_k^T + N_k)^{-1} H_k \Gamma_k) \\ &= K_k H_k \Gamma_k \end{aligned} \tag{31}$$

So, given a belief state $\widehat{\boldsymbol{x}}_k, P_k$ and control input $\boldsymbol{u}_k$, but before a measurement sampled from the distribution of $\boldsymbol{y}_{k+1}$ is observed, the mean of the belief state $\widehat{\boldsymbol{x}}_{k+1}$ given by the belief update is stochastic and may expressed by the following equation:

$$\widehat{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w}_k, \qquad \boldsymbol{w}_k \sim \mathcal{N}\left[\boldsymbol{0}, W[\widehat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]\right] \tag{32}$$

where

$$W[\widehat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = K_k H_k \Gamma_k \tag{33}$$

Note that $W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ is the variance of the mean $\hat{\boldsymbol{x}}_{k+1}$ given by the belief update before a measurement is taken, *not* the variance P_{k+1} that parameterizes the belief state.

The variance that parameterizes the belief state is given by

$$P_{k+1} = \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \tag{34}$$

where,

$$\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = (\Gamma_k^{-1} + H_k^T N_k^{-1} H_k)^{-1} \tag{35}$$

which was given by Equation 20. This implies that the variance dynamics are deterministic. Therefore, given a previous belief state $\hat{\boldsymbol{x}}_k, P_k$ and control input $\boldsymbol{u}_k$, the variance of the belief state after a belief update is always the same value regardless of what measurement is made. This is in contrast to the mean dynamics, which are stochastic, i.e. given a previous belief state $\hat{\boldsymbol{x}}_k, P_k$ and control input $\boldsymbol{u}_k$, the mean $\hat{\boldsymbol{x}}_{k+1}$ of the belief state after the belief update depends on which measurement is made, which is uncertain prior to receiving the actual measurement. Once a measurement is made, the

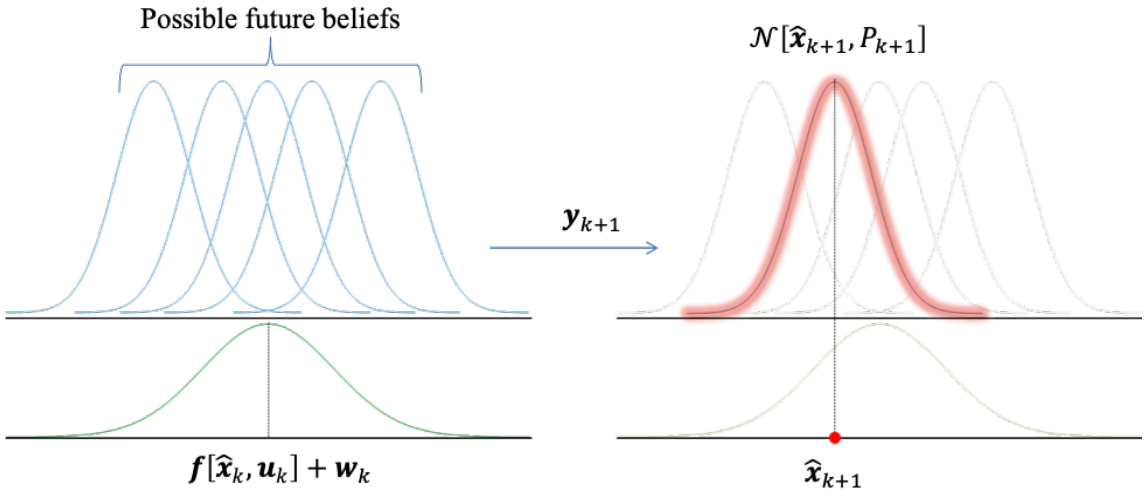


Figure 9: Before the measurement $\boldsymbol{y}_{k+1}$ is observed, the mean $\hat{\boldsymbol{x}}_{k+1}$ of the succeeding belief state is uncertain, and is Gaussian-distributed with variance $W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$. After the measurement $\boldsymbol{y}_{k+1}$ is observed, the mean $\hat{\boldsymbol{x}}_{k+1}$ the succeeding belief state collapses to a unique value.

distribution of the belief state collapses to a single belief state by using the measured value of $\boldsymbol{y}_{k+1}$ in the belief update (Equations 19-24). The belief dynamics describe a Gaussian distribution over the means of Gaussian distributions with the same variance, which collapses to a belief state that is Gaussian with known mean and variance once an actual measurement is made. This is illustrated in Figure 9.

4.4 VALUE ITERATION

The value function $v_k[\hat{\boldsymbol{x}}_k, P_k]$ at time k is approximated by a function that is quadratic in the mean and linear in the variance, that is locally valid around some nominal belief $\overline{\boldsymbol{x}}_k, \bar{P}_k$:

$$v_k[\hat{\boldsymbol{x}}_k, P_k] \approx s_k + \frac{1}{2}(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k)^T S_k(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + \boldsymbol{s}_k^T(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + \mathrm{tr}[T_k(P_k - \bar{P}_k)] \quad (36)$$

with $S_k \geq 0$. The form of this approximation is natural for cost functions that are quadratic in the state, since the expected cost is then quadratic with respect to the mean and linear with respect to the variance from the identity

$$\mathrm{E}[\boldsymbol{z}^T A \boldsymbol{z}] = \mathrm{E}[\boldsymbol{z}^T] A \mathrm{E}[\boldsymbol{z}] + \mathrm{tr}\big[A\ \mathrm{Var}[\boldsymbol{z}]\big], \quad (37)$$

as noted in [118].

4.4.1 VALUE FUNCTION APPROXIMATIONS

Suppose that an approximation of the value function $v_{k+1}[\hat{\boldsymbol{x}}_{k+1}, P_{k+1}]$ of the form given in Equation 36 exists at time step $k+1$, with parameters s_{k+1}, $\boldsymbol{s}_{k+1}^T$, S_{k+1}, T_{k+1} and nominal belief $\overline{\boldsymbol{x}}_{k+1}$, $\bar{P}_{k+1}$. It will now be shown that an approximation of the value function of the form given in Equation 36 exists at time step k through linearizing the belief dynamics and quadratizing the immediate reward function, so that parameters s_k,

$\boldsymbol{s}_k^T$, S_k, T_k and nominal belief $\overline{\boldsymbol{x}}_k$, $\bar{P}_k$ may be determined at time step k if the parameters and nominal belief are given at time $k+1$.

From the Bellman backup (Equation 4), it necessary to find the control input which minimizes (rather than maximizes, as noted in Section 4.1) the expected value of the value function at each belief.

$$v_k[\hat{\boldsymbol{x}}, P] = \min_{\boldsymbol{u}}\left(c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + \mathrm{E}\left[v_{k+1}[\hat{\boldsymbol{x}}_{k+1}, P_{k+1}]\right]\right) \tag{38}$$

From the approximation of the value function given by Equation 36 and the belief state dynamics given by Equations 32-35, Equation 38 may be approximated as

$$\begin{aligned} v_k[\hat{\boldsymbol{x}}_k, P_k] \approx \min_{\boldsymbol{u}_k}\Bigg(& c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + \mathrm{E}\Bigg[s_{k+1} \\ &+ \frac{1}{2}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w}_k - \overline{\boldsymbol{x}}_{k+1})^T S_{k+1}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w}_k - \overline{\boldsymbol{x}}_{k+1}) \\ &+ \boldsymbol{s}_{k+1}^T(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w}_k - \overline{\boldsymbol{x}}_{k+1}) \\ &+ \mathrm{tr}[T_k(\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1})]\Bigg]\Bigg) \\ \approx \min_{\boldsymbol{u}_k}\Bigg(& c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + s_{k+1} \\ &+ \frac{1}{2}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1})^T S_{k+1}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1}) \\ &+ \boldsymbol{s}_{k+1}^T(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1}) + \mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1})] \\ &+ \frac{1}{2}\mathrm{tr}[S_{k+1}W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k])]\Bigg) \end{aligned} \tag{39}$$

where last term comes from applying the identity in Equation 37 to the expectation of the term which is quadratic in $\boldsymbol{w}_k$. This will further be approximated by linearizing the belief dynamics about a nominal belief $\overline{\boldsymbol{x}}_k$, $\bar{P}_k$ and control input $\overline{\boldsymbol{u}}_k$ that are selected such that

$$\bar{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k] \tag{40}$$

$$\bar{P}_{k+1} = \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \tag{41}$$

Remaining consistent with the format of Equations 40-41, the over-bar (e.g. $\bar{\boldsymbol{z}}$) will be used to denote declared variables that are calculated using the nominal values $\bar{\boldsymbol{x}}_k, \bar{P}_k$, and $\bar{\boldsymbol{u}}_k$.

The deterministic part of the mean dynamics may be linearized as

$$\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \bar{\boldsymbol{x}}_{k+1} \approx \bar{F}_k(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \bar{G}_k(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) \tag{42}$$

where,

$$\bar{F}_k = \frac{\partial \boldsymbol{f}}{\partial \hat{\boldsymbol{x}}_k}[\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k], \qquad \bar{G}_k = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}[\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k] \tag{43}$$

The method used here to linearize the dynamics of the variance (second to last term in final equation of Equation 39) and the stochastic part of the mean dynamics (last term of Equation 39) is what distinguishes this method from the method used by [118]. In [118], the traces of matrix products in Equation 39 were represented as the dot product of two vectorized $n \times n$ matrices (see Appendix A). This made it convenient to perform linearization, although it was among the sources of greatest computational cost in the algorithm. Instead of employing vectorization, the linearization can equivalently be performed though the use of directional derivatives [106]. The following properties of matrix derivatives will also be used [75]

$$\frac{\partial A^{-1}}{\partial z} = -A^{-1}\frac{\partial A}{\partial z}A^{-1} \tag{44}$$

$$\frac{\partial \mathrm{tr}[A]}{\partial z} = \mathrm{tr}\left[\frac{\partial A}{\partial z}\right] \tag{45}$$

First, linearization of the variance of the belief state will be performed by noting that

$$\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1} = \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]. \tag{46}$$

For the sake of clarity, the first order Taylor series expansions of the mean, variance, and control input will all be considered separately by assuming that the parameters that are not included in each expansion are held fixed (justification for this is provided in Appendix C). The first order Taylor series expansion of $\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]$ is then given by the sum of the individual expansions.

$$\begin{aligned} &\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k] \\ &\qquad \approx \ (\Phi[\hat{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]) \\ &\qquad + (\Phi[\overline{\boldsymbol{x}}_k, P_k, \overline{\boldsymbol{u}}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]) \\ &\qquad + (\Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]) \end{aligned} \tag{47}$$

Additionally, the properties of traces and the above equation imply the following about the second to last term in Equation 47,

$$\begin{aligned} &\mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k])] \\ &\qquad \approx \mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k])] \\ &\qquad + \mathrm{tr}[T_{k+1}(\Phi[\overline{\boldsymbol{x}}_k, P_k, \overline{\boldsymbol{u}}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k])] \\ &\qquad + \mathrm{tr}[T_{k+1}(\Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \boldsymbol{u}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k])] \end{aligned} \tag{48}$$

The linearization of $\Phi[\overline{\boldsymbol{x}}_k, P_k, \overline{\boldsymbol{u}}_k] - \Phi[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]$ will be performed first. Whereas van den Berg et al. [118] proposed taking the first order Taylor series expansion with respect to each element of P_k (see Appendix A), here it is proposed that it is more efficient to take the first order Taylor series expansion of a directional derivative. First, let P_k be represented as a deviation ΔP_k from the nominal value $\bar{P}_k$.

$$P_k = \bar{P}_k + \Delta P_k \tag{49}$$

Now, let δP_k be a scalar multiple α of ΔP_k.

$$\Delta P_k = \alpha \delta P_k \tag{50}$$

If the elements of ΔP_k and δP_k were arranged into a vector, it can be seen that ΔP_k and δP_k would be in the same direction, since they vary only by a scaling factor α. Taking the first order Taylor series expansion of $\Phi[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$ about $\alpha = 0$ yields

$$\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] \approx \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] + \left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial \alpha}\right|_{\alpha=0} (\alpha - 0) \tag{51}$$

This expansion is valid for any choice of δP_k. To evaluate the partial derivative in Equation 51, first allow $\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]$ to be represented as the inverse of its own inverse.

$$\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] = (\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]^{-1})^{-1} \tag{52}$$

This allows for the application of the identity given in Equation 44,

$$\left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial \alpha}\right|_{\alpha=0} = -\bar{P}_{k+1} \left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]^{-1}}{\partial \alpha}\right|_{\alpha=0} \bar{P}_{k+1} \tag{53}$$

The partial derivative of $\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]^{-1}$ in Equation 53 may be found by referencing Equation 20.

$$\begin{aligned} \left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]^{-1}}{\partial \alpha}\right|_{\alpha=0} &= \frac{\partial}{\partial \alpha}(\Gamma_k^{-1} + H_k^T N_k^{-1} H_k)\Big|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}} \\ &= \left.\frac{\partial \Gamma_k^{-1}}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}} \end{aligned} \tag{54}$$

since Γ_k^{-1} is a function of α, which can be seen from Equations 21 and 49-50, while the second term is not. By once again applying the identity in Equation 44 and evaluating at $\alpha = 0$, the following is obtained:

$$\left.\frac{\partial \Gamma_k^{-1}}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k,\alpha=0,\bar{\boldsymbol{u}}} = -\bar{\Gamma}_k^{-1}\left.\frac{\partial \Gamma_k}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k,\alpha=0,\bar{\boldsymbol{u}}}\bar{\Gamma}_k^{-1} \tag{55}$$

From Equation 21, it can be seen that the partial derivative in the above equation is

$$\left.\frac{\partial \Gamma_k}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k,\alpha=0,\bar{\boldsymbol{u}}} = \bar{F}_k\left.\frac{\partial P_k}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k,\alpha=0,\bar{\boldsymbol{u}}}\bar{F}_k^T \tag{56}$$

since F_k and M_k are not a function of α. Expressing P_k once again as a deviation from the nominal value $\bar{P}_k$, as given Equation 49, and evaluating at $\alpha = 0$ yields

$$\bar{F}_k\left.\frac{\partial(\bar{P}_k + \alpha\delta P_k)}{\partial \alpha}\right|_{\bar{\boldsymbol{x}}_k,\alpha=0,\bar{\boldsymbol{u}}}\bar{F}_k^T = \bar{F}_k\delta P_k\bar{F}_k^T \tag{57}$$

Substituting the results of Equations 53-57 into Equation 51 yields

$$\left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial \alpha}\right|_{\alpha=0} = \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1} \tag{58}$$

Substituting the above equation into the Taylor series expansion in Equation 51 yields,

$$\begin{aligned}\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] &\approx (\bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1})(\alpha - 0)\\ &\approx \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k(\alpha\delta P_k)\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\end{aligned} \tag{59}$$

where multiplication by α is commutative since it is scalar. From Equation 50, it can be seen that ΔP_k may be substituted into the above equation.

$$\Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \approx \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\Delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1} \tag{60}$$

Now, since δP_k was chosen arbitrarily, this approximation is valid for any matrix ΔP_k. Equation 60 may then be substituted into the second to last term in Equation 48 to give:

$$\begin{aligned}\mathrm{tr}[T_{k+1}(\Phi[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] &\approx \mathrm{tr}[T_{k+1}\bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\Delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}]\\ &\approx \mathrm{tr}[\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}T_{k+1}\bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\Delta P_k]\end{aligned} \tag{61}$$

where the cyclic property of traces was used to rearrange the product inside the trace. Now, let X_k be defined as the matrix product that precedes ΔP_k in Equation 61.

$$X_k = \bar{F}_k^T \bar{\Gamma}_k^{-1} \bar{P}_{k+1} T_{k+1} \bar{P}_{k+1} \bar{\Gamma}_k^{-1} \bar{F}_k \tag{62}$$

A form similar to the second to last term in Equation 39 can be obtained by referencing Equations 61-62.

$$\mathrm{tr}[T_{k+1}(\Phi[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \approx \mathrm{tr}[X_k(P_k - \bar{P}_k)] \tag{63}$$

Now the linearization of the term $\Phi[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$ will be performed. While the preceding Taylor series expansion employed the directional derivative, it is usually not as convenient to apply here. Instead, the partial derivatives with respect to each element $\hat{x}_i$ of $\hat{\boldsymbol{x}}_k$ are employed in the expansion about $\bar{\boldsymbol{x}}_k$.

$$\Phi[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \approx \sum_i \left[\left. \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i} \right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} (\hat{x}_i - \bar{x}_i) \right] \tag{64}$$

where $\bar{x}_i$ is an element of $\bar{\boldsymbol{x}}_k$. The trace of this quantity is then

$$\begin{aligned} &\mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \\ &\qquad \approx \mathrm{tr}\left[T_{k+1} \left(\sum_i \left. \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i} \right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} (\hat{x}_i - \bar{x}_i) \right) \right] \\ &\qquad \approx \sum_i \left(\mathrm{tr}\left[T_{k+1} \left. \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i} \right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] (\hat{x}_i - \bar{x}_i) \right) \end{aligned} \tag{65}$$

which may be derived from the properties of traces, and since $(\hat{x}_i - \bar{x}_i)$ is scalar. Note that Equation 65 may be expressed as an inner product of two vectors

$$\mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \approx \boldsymbol{a}_k^T (\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}_k) \tag{66}$$

where an element a_i of $\boldsymbol{a}_k$ is given by

$$a_i = \mathrm{tr}\left[T_{k+1} \left. \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i} \right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] \tag{67}$$

The partial derivative in Equation 67 may be found numerically, or it may be found analytically using the methods presented in Appendix B.

The process for linearization with respect to the control input is very similar to the linearization with respect to the mean. The resulting approximation of the third term in Equation 48 is

$$\begin{aligned}
&\mathrm{tr}[T_{t+1}(\Phi[\bar{\boldsymbol{x}}_t, \bar{P}_t, \boldsymbol{u}_t] - \Phi[\bar{\boldsymbol{x}}_t, \bar{P}_t, \bar{\boldsymbol{u}}_t])] \\
&\qquad \approx \sum_i \left(\mathrm{tr}\left[T_{k+1} \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial u_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] (u_i - \bar{u}_i) \right) \qquad (68) \\
&\qquad \approx \boldsymbol{b}_k^T (\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)
\end{aligned}$$

where an element b_i of $\boldsymbol{b}_k$ is given by

$$b_i = \mathrm{tr}\left[T_{k+1} \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial u_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] \qquad (69)$$

Again, the partial derivative in Equation 69 may be found numerically, or it may be found analytically using the methods presented in Appendix B.

The linearization of $W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ can be performed similarly to the linearization of the variance of the belief state. First, note that the first order Taylor series expansion of $W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$ can be represented as the sum of the individual expansions.

$$\begin{aligned}
W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \approx\ & W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] + (W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]) \\
&+ (W[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]) \qquad (70) \\
&+ (W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \boldsymbol{u}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])
\end{aligned}$$

The properties of traces and the above equation imply the last term in Equation 39 may be approximated as

$$
\begin{aligned}
&\mathrm{tr}\left[S_{k+1}W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]\right] \\
&\qquad \approx \mathrm{tr}\left[S_{k+1}W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]\right] \\
&\qquad + \mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \\
&\qquad + \mathrm{tr}[S_{k+1}(W[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \\
&\qquad + \mathrm{tr}[S_{k+1}(W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \boldsymbol{u}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])]
\end{aligned} \tag{71}
$$

The linearization of $W[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$ will be performed first. The assumptions of Equations 49-50 are used to form the Taylor series expansion of this term about $\alpha = 0$.

$$
W[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \approx \left.\frac{\partial W[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial\alpha}\right|_{\alpha=0} (\alpha - 0) \tag{72}
$$

Using Equation 33, the partial derivative in Equation 72 can be evaluated as

$$
\begin{aligned}
\left.\frac{\partial W[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial\alpha}\right|_{\alpha=0} &= \left.\frac{\partial}{\partial\alpha} K_k H_k \Gamma_k\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}_k} \\
&= \left.\frac{\partial K_k}{\partial\alpha}\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}_k} \bar{H}_k \bar{\Gamma}_k + \bar{K}_k \bar{H}_k \left.\frac{\partial \Gamma_k}{\partial\alpha}\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}_k}
\end{aligned} \tag{73}
$$

The first partial derivative in Equation 73 is given by,

$$
\begin{aligned}
\left.\frac{\partial K_k}{\partial\alpha}\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}_k} &= \left.\frac{\partial \Phi[\bar{\boldsymbol{x}}_k, P_k, \bar{\boldsymbol{u}}_k]}{\partial\alpha} H_k^T N_k^{-1}\right|_{\bar{\boldsymbol{x}}_k, \alpha=0, \bar{\boldsymbol{u}}_k} \\
&= \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k \delta P_k \bar{F}_k^T \bar{\Gamma}_k^{-1} \bar{P}_{k+1} \bar{H}_k^T \bar{N}_k^{-1}
\end{aligned} \tag{74}
$$

which is obtained by applying Equations 22 and 58 and evaluating at the nominal values. The second partial derivative in Equation 73 is given by Equations 56-57. Substitution of Equations 74 and 56-57 into Equation 73 yields

$$\left.\frac{\partial W[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k]}{\partial \alpha}\right|_{\alpha=0} \tag{75}$$
$$= \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\bar{H}_k^T\bar{N}_k^{-1}\bar{H}_k\bar{\Gamma}_k + \bar{K}_k\bar{H}_k\bar{F}_k\delta P_k\bar{F}_k^T$$

Substituting Equation 75 into Equation 72 gives

$$\begin{aligned} &W[\bar{\boldsymbol{x}}_k, \bar{P}_k + \alpha\delta P_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \\ &\quad\approx (\bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\bar{H}_k^T\bar{N}_k^{-1}\bar{H}_k\bar{\Gamma}_k + \bar{K}_k\bar{H}_k\bar{F}_k\delta P_k\bar{F}_k^T)\alpha \\ &\quad\approx \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k(\alpha\delta P_k)\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\bar{H}_k^T\bar{N}_k^{-1}\bar{H}_k\bar{\Gamma}_k + \bar{K}_k\bar{H}_k\bar{F}_k(\alpha\delta P_k)\bar{F}_k^T \\ &\quad\approx \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k\Delta P_k\bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\bar{H}_k^T\bar{N}_k^{-1}\bar{H}_k\bar{\Gamma}_k + \bar{K}_k\bar{H}_k\bar{F}_k\Delta P_k\bar{F}_k^T \end{aligned} \tag{76}$$

which obtained by distributing α and applying Equation 50. It is convenient to represent the products before and after the ΔP_k values as single terms.

$$\begin{aligned} \Psi_1 &= \bar{F}_k^T\bar{\Gamma}_k^{-1}\bar{P}_{k+1}\bar{H}_k^T\bar{N}_k^{-1}\bar{H}_k\bar{\Gamma}_k \\ \Psi_2 &= \bar{P}_{k+1}\bar{\Gamma}_k^{-1}\bar{F}_k \\ \Psi_3 &= \bar{F}_k^T \\ \Psi_4 &= \bar{K}_k\bar{H}_k\bar{F}_k \end{aligned} \tag{77}$$

Substituting Equations 76-77 into the second to last term in Equation 71 yields

$$\begin{aligned} \mathrm{tr}[S_{k+1}(W[\bar{\boldsymbol{x}}_k, P, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] &\approx \mathrm{tr}[S_{k+1}(\Psi_2\Delta P_k\Psi_1 + \Psi_4\Delta P_k\Psi_3)] \\ &\approx \mathrm{tr}[(\Psi_1 S_{k+1}\Psi_2 + \Psi_3 S_{k+1}\Psi_4)\Delta P_k] \end{aligned} \tag{78}$$

where the properties of the sum of traces and the cyclic property of traces were used to rearrange the product inside the trace. Now, let V_k be defined as the matrix product that precedes ΔP_k in Equation 78:

$$V_k = \Psi_1 S_{k+1}\Psi_2 + \Psi_3 S_{k+1}\Psi_4 \tag{79}$$

A form similar to the second to last term in Equation 39 can be obtained by referencing Equations 78-79.

$$\mathrm{tr}[S_{k+1}(W[\bar{\boldsymbol{x}}_k, P, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \approx \mathrm{tr}[V_k(P_k - \bar{P}_k)] \tag{80}$$

Although it is not shown here, the matrix V_k also happens to be symmetric.

The process for linearization with respect to the mean and the control input is very similar to the linearization performed in Equation 65. The resulting approximations of the second and fourth terms in Equation 71 are

$$\mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])]$$

$$\approx \sum_i \left(\mathrm{tr}\left[S_{k+1} \frac{\partial W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] (\hat{x}_i - \bar{x}_i) \right) \tag{81}$$

$$\mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])]$$

$$\approx \sum_i \left(\mathrm{tr}\left[S_{k+1} \frac{\partial W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial u_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] (u_i - \bar{u}_i) \right) \tag{82}$$

which may be expressed as the inner vector products

$$\mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \approx \boldsymbol{t}_k^T(\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}_k) \tag{83}$$

$$\mathrm{tr}[S_{k+1}(W[\hat{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] - W[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k])] \approx \boldsymbol{v}_k^T(\boldsymbol{u} - \bar{\boldsymbol{u}}_k) \tag{84}$$

where elements t_i of $\boldsymbol{t}_k$ and v_i of $\boldsymbol{v}_k$ are given by

$$t_i = \mathrm{tr}\left[S_{k+1} \frac{\partial W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial \hat{x}_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] \tag{85}$$

$$v_i = \mathrm{tr}\left[S_{k+1} \frac{\partial W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial u_i}\bigg|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \right] \tag{86}$$

The partial derivatives in Equations 85-86 may be found numerically, or they may be found analytically using the methods presented in Appendix B.

The term $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ in Equation 39 is approximated using a second order Taylor series expansion about the nominal values $\overline{\boldsymbol{x}}_k$ and $\overline{\boldsymbol{u}}_k$, and a first order Taylor series expansion with about $\bar{P}_k$:

$$c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \approx q_k + \frac{1}{2}\begin{bmatrix}\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix}^T \begin{bmatrix}Q_k & J_k^T \\ J_k & R_k\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix} + \begin{bmatrix}\boldsymbol{q}_k \\ \boldsymbol{r}_k\end{bmatrix}^T \begin{bmatrix}\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix} + \mathrm{tr}[U_k(P_k - \bar{P}_k)] \tag{87}$$

where

$$\begin{aligned} Q_k &= \frac{\partial^2 c_k}{\partial \hat{\boldsymbol{x}}_k \partial \hat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & \boldsymbol{q}_k^T &= \frac{\partial c_k}{\partial \hat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\ R_k &= \frac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & \boldsymbol{r}_k^T &= \frac{\partial c_k}{\partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\ J_k &= \frac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \hat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & q_k &= c_k[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\ U_k &= \frac{\partial c_k}{\partial P_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \end{aligned} \tag{88}$$

where either directional derivatives or element-wise differentiation may be used to obtain U_k, whichever is more convenient.

4.4.2 BELLMAN BACKUP SUMMARY

The Bellman backup equation may now be approximated using the approximations derived in Section 4.4.1. First, recall the Bellman backup equation:

$$
\begin{aligned}
v_k[\hat{\boldsymbol{x}}_k, P_k] \approx \min_{\boldsymbol{u}_k} \Big(& c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + s_{k+1} \\
& + \frac{1}{2}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \bar{\boldsymbol{x}}_{k+1})^T S_{k+1}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \bar{\boldsymbol{x}}_{k+1}) \\
& + \boldsymbol{s}_{k+1}^T(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \bar{\boldsymbol{x}}_{k+1}) + \mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1})] \\
& + \frac{1}{2}\mathrm{tr}[S_{k+1} W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k])] \Big)
\end{aligned}
\tag{89}
$$

The following approximations summarize the results of Section 4.4.1:

$$
\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \bar{\boldsymbol{x}}_{k+1} \approx \bar{F}_k(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \bar{G}_k(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) \tag{90}
$$

$$
\begin{aligned}
& \mathrm{tr}[T_{k+1}(\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1})] \\
& \qquad \approx \mathrm{tr}[X_k(P_k - \bar{P}_k)] + \boldsymbol{a}_k^T(\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}_k) + \boldsymbol{b}_k^T(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)
\end{aligned}
\tag{91}
$$

$$
\mathrm{tr}[S_{k+1} W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k])] \approx w_k + \mathrm{tr}[V_k(P_k - \bar{P}_k)] + \boldsymbol{t}_k^T(\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}_k) + \boldsymbol{v}_k^T(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) \tag{92}
$$

$$
\begin{aligned}
c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \approx q_k & + \frac{1}{2}\begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix}^T \begin{bmatrix} Q_k & J_k^T \\ J_k & R_k \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{q}_k \\ \boldsymbol{r}_k \end{bmatrix}^T \begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix} \\
& + \mathrm{tr}[U_k(P_k - \bar{P}_k)]
\end{aligned}
\tag{93}
$$

Now, Equations 90-93 can be substituted into Equation 89, which results in the following after like terms are collected and the result presented in matrix form

$$
\begin{aligned}
v_k[\hat{\boldsymbol{x}}_k, P_k] \approx e_k & + \frac{1}{2}\begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix}^T \begin{bmatrix} C_k & E_k^T \\ E_k & D_k \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{c}_k \\ \boldsymbol{d}_k \end{bmatrix}^T \begin{bmatrix} \hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \end{bmatrix} \\
& + \mathrm{tr}[Y_k(P_k - \bar{P}_k)]
\end{aligned}
\tag{94}
$$

where

$$
\begin{aligned}
C_k &= Q_k + \bar{F}_k^T S_{k+1} \bar{F}_k, & D_k &= R_k + \bar{G}_k^T S_{k+1} \bar{G}_k, \\
E_k &= J_k + \bar{G}_k^T S_{k+1} \bar{F}_k, & Y_k &= U_k + X_k + \frac{1}{2} V_k, \\
\boldsymbol{c}_k^T &= \boldsymbol{q}_k^T + \boldsymbol{s}_{k+1}^T \bar{F}_k + \boldsymbol{a}_k^T + \frac{1}{2}\boldsymbol{t}_k^T, & \boldsymbol{d}_k^T &= \boldsymbol{r}_k^T + \boldsymbol{s}_{k+1}^T \bar{G}_k + \boldsymbol{b}_k^T + \frac{1}{2}\boldsymbol{v}_k^T,
\end{aligned}
\tag{95}
$$

$$e_k = q_k + s_{k+1} + \frac{1}{2} w_k,$$

A locally optimal policy $\boldsymbol{u}_k = \boldsymbol{\pi}_k[\hat{\boldsymbol{x}}_k, P_k]$ at time step k can be found by referencing Equations 94-95 and setting the first derivative of $v_k[\hat{\boldsymbol{x}}_k, P_k]$ with respect to $\boldsymbol{u}_k$ equal to zero and then solving for $\boldsymbol{u}_k$.

$$\boldsymbol{u}_k = L_k(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \boldsymbol{l}_k + \bar{\boldsymbol{u}}_k \tag{96}$$

where

$$L_k = -D_k^{-1} E_k \tag{97}$$

$$\boldsymbol{l}_k = -D_k^{-1} \boldsymbol{d}_k \tag{98}$$

Note that D_k is invertible since it was required that $R_k > 0$ and $S_{k+1} \geq 0$. By substitution of Equations 96-98 into Equation 94, the desired form of the value function approximation is obtained at time step k.

$$\begin{aligned} v_k[\hat{\boldsymbol{x}}_k, P_k] \approx s_k + \frac{1}{2}(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k)^T S_k (\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \boldsymbol{s}_k^T (\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) \\ + \operatorname{tr}[T_k (P_k - \bar{P}_k)] \end{aligned} \tag{99}$$

where

$$\begin{aligned} S_k &= C_k + L_k^T E_k, & \boldsymbol{s}_k^T &= \boldsymbol{c}_k^T + \boldsymbol{l}_k^T E_k, \\ s_k &= e_k + \frac{1}{2} \boldsymbol{d}_k^T \boldsymbol{l}_k, & T_k &= Y_k, \end{aligned} \tag{100}$$

Therefore, given a value function $v_{k+1}[\hat{\boldsymbol{x}}_{k+1}, P_{k+1}]$ of the form assumed in Equation 36 and given that it is possible to select $\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k$ such that $\bar{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k]$ and $\bar{P}_{k+1} = \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$, then $v_k[\hat{\boldsymbol{x}}_k, P_k]$ may be approximated using Equations 99-100.

4.4.3 ITERATING TO A LOCALLY-OPTIMAL POLICY

Sections 4.4.1 and 4.4.2 assume that it possible to select $\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k$ such that $\bar{\boldsymbol{x}}_{k+1} = \boldsymbol{f}[\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k]$ and $\bar{P}_{k+1} = \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k]$ in order to generate the approximation of $v_k[\hat{\boldsymbol{x}}_k, P_k]$ from the approximation of $v_{k+1}[\hat{\boldsymbol{x}}_{k+1}, P_{k+1}]$. This is most easily accomplished by assuming a policy, which is applied to the system beginning at a given initial belief state $\bar{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0,\ \bar{P}_0 = P_0$. Then, successive values of $\bar{\boldsymbol{x}}_{k+1}$ and $\bar{P}_{k+1}$ are generated by applying the policy and Equations 40-41. This is continued until the length of the planning horizon, ℓ, is reached. This process thus generates a sequence of nominal beliefs and actions which satisfy Equations 40-41 : $\{(\bar{\boldsymbol{x}}_0, \bar{P}_0, \bar{\boldsymbol{u}}_0), \dots, (\bar{\boldsymbol{x}}_{\ell-1}, \bar{P}_{\ell-1}, \bar{\boldsymbol{u}}_{\ell-1}), (\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell)\}$.

By approximating the value function at the final time step $k = \ell$, $v_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$, it is then possible to apply back-propagation as described in Section 4.4.2 to find an approximately optimal value function $v_k[\hat{\boldsymbol{x}}_k, P_k]$ and policy $\boldsymbol{\pi}_k[\hat{\boldsymbol{x}}_k, P_k]$ for each time step k. The value function $v_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$ is approximated by using a second order Taylor series expansion of $c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$ about the nominal belief $\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell$:

$$\begin{aligned} S_\ell &= \frac{\partial^2 c_\ell}{\partial \hat{\boldsymbol{x}}_\ell \partial \hat{\boldsymbol{x}}_\ell}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], & \boldsymbol{s}_\ell^T &= \frac{\partial c_\ell}{\partial \hat{\boldsymbol{x}}_\ell}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], \\ s_\ell &= c_\ell[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], & T_\ell &= \frac{\partial c_\ell}{\partial P_\ell}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], \end{aligned} \tag{101}$$

where the directional derivative or element-wise differentiation may be used to obtain T_ℓ, whichever is more convenient.

An initial policy (or simply a sequence of actions) is assumed in order to generate an initial nominal trajectory $\left\{\left(\bar{\boldsymbol{x}}_0^{(0)}, \bar{P}_0^{(0)}, \bar{\boldsymbol{u}}_0^{(0)}\right), \dots, \left(\bar{\boldsymbol{x}}_{\ell-1}^{(0)}, \bar{P}_{\ell-1}^{(0)}, \bar{\boldsymbol{u}}_{\ell-1}^{(0)}\right), \left(\bar{\boldsymbol{x}}_\ell^{(0)}, \bar{P}_\ell^{(0)}\right)\right\}$, where $\bar{\boldsymbol{u}}_k^{(0)}$ is the action that was applied at belief $\bar{\boldsymbol{x}}_k^{(0)}, \bar{P}_k^{(0)}$, and the initial nominal belief is

equal to the given initial belief: $\bar{\boldsymbol{x}}_0^{(0)} = \hat{\boldsymbol{x}}_0$, $\bar{P}_0^{(0)} = P_0$. The initial policy affects the convergence to a locally optimal policy, and so it may require careful selection to obtain the desired results. Then, given a nominal trajectory at the iteration $i-1$, the nominal trajectory at the iteration $i-1$ can be used to find an approximately optimal value function $v_k^{(i-1)}[\hat{\boldsymbol{x}}_k, P_k]$ and locally optimal policy given by Equation 102 (which is based on Equation 96).

$$\boldsymbol{u}_k = L_k^{(i-1)}\left(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k^{(i-1)}\right) + \boldsymbol{l}_k^{(i-1)} + \bar{\boldsymbol{u}}_k^{(i-1)}. \tag{102}$$

Since the policy given in Equation 102 is locally optimal, it is expected to output a locally optimal action for any mean $\hat{\boldsymbol{x}}_k$ in the neighborhood of $\bar{\boldsymbol{x}}_k^{(i-1)}$. Thus, the policy given in Equation 102 may be expected to improve upon the previous policy[7], so that a lower total expected cost can be expected from applying the policy in Equation 102 compared to the previous policy. Therefore, the policy given in Equation 102 it is used to generate a new nominal trajectory for iteration i:

$$\bar{\boldsymbol{u}}_k^{(i)} = L_k^{(i-1)}\left(\bar{\boldsymbol{x}}_k^{(i)} - \bar{\boldsymbol{x}}_k^{(i-1)}\right) + \boldsymbol{l}_k^{(i-1)} + \bar{\boldsymbol{u}}_k^{(i-1)} \tag{103}$$

$$\bar{\boldsymbol{x}}_{k+1}^{(i)} = \boldsymbol{f}\left[\bar{\boldsymbol{x}}_k^{(i)}, \bar{\boldsymbol{u}}_k^{(i)}\right] \tag{104}$$

$$\bar{P}_{k+1}^{(i)} = \Phi\left[\bar{\boldsymbol{x}}_k^{(i)}, \bar{P}_k^{(i)}, \bar{\boldsymbol{u}}_k^{(i)}\right] \tag{105}$$

where $\bar{\boldsymbol{x}}_0^{(i)} = \hat{\boldsymbol{x}}_0$, $\bar{P}_0^{(i)} = P_0$. Once a new nominal trajectory is computed, it is possible to compute a new value function $v_k^{(i)}[\hat{\boldsymbol{x}}_k, P_k]$ and policy and then iterate. It has been shown that the policy causes convergence to a locally optimal trajectory with a second-order convergence rate [118].

[7] Since it is a locally optimal policy, it should dominate any policy in its neighborhood.

The policy described in Equation 102 is only valid in the neighborhood the nominal belief $\bar{\boldsymbol{x}}_k^{(i-1)}$, $\bar{P}_k^{(i-1)}$. Therefore, if the nominal belief $\bar{\boldsymbol{x}}_k^{(i)}$, $\bar{P}_k^{(i)}$ is too different from $\bar{\boldsymbol{x}}_k^{(i-1)}$, $\bar{P}_k^{(i-1)}$, then the policy in Equation 102 may not select a good action, which in turn may lead to higher total expected cost than the previous iteration. To mitigate this issue, van den Berg et al. [118] suggest augmenting this algorithm with line search such that a candidate trajectory is only accepted if it has lower total expected cost than the current nominal trajectory by introducing the parameter ε.

$$\bar{\boldsymbol{u}}_k^{(i)} = L_k^{(i-1)}\left(\bar{\boldsymbol{x}}_k^{(i)} - \bar{\boldsymbol{x}}_k^{(i-1)}\right) + \varepsilon \boldsymbol{l}_k^{(i-1)} + \bar{\boldsymbol{u}}_k^{(i-1)} \tag{106}$$

If the total expected cost of the candidate trajectory is greater than the cost of the current nominal trajectory, ε is divided and half and a new candidate trajectory is computed. If the candidate trajectory is accepted, the candidate trajectory becomes the new nominal trajectory, ε is reset to 1, and value iteration continues. Since it is costly to perform back-propagation using the method in Section 4.4.2, an approximation be found efficiently by computing the expected cost of the candidate trajectory with respect to the control policy of the current nominal trajectory. This avoids the need to recompute the values in Equations 95 and 100 in order to evaluate the expected cost of the candidate trajectory. More details on this approach are given in [118]. However, a different approach was employed in this work.

In this work, an artificial cost R'_k was introduced to regulate the magnitude of the terms which are added to the nominal control input in Equation 103. This was accomplished by imposing the cost R'_k on the deviation of the input from the nominal value.

$$c'_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + \frac{1}{2}(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)^T R'_k(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) \quad (107)$$

The net result is that a damping factor is introduced into the value of D_k in Equation 95, similar to what is used in the Levenberg-Marquardt algorithm [65]. This parameter was then manually tuned to optimize the rate of convergence while minimizing the probability the nominal trajectory would diverge, which eliminated the need for estimating the total expected cost and generating new candidate trajectories. This method is *ad hoc* and may not work well for other systems. The tuning of R'_k is discussed in Section 6.2.1.1.

4.5 COMPLEXITY ANALYSIS

The complexity of the iLQG algorithm will be analyzed in terms of the size of the state space, n, and the length of the time horizon, ℓ. In this analysis, it is assumed that the multiplication of a $p \times q$ matrix and a $q \times r$ requires $O[pqr]$ time to compute and that the inversion of a $n \times n$ matrix requires $O[n^3]$ time. The complexity of the iLQG algorithm presented here is dependent on the size of the input and measurement spaces, and on the complexity of the functions $\boldsymbol{f}$, $\boldsymbol{h}$, M, N, and c_k. For the sake of comparison, the sizes of these spaces and the complexities of these functions are assumed based on the assumptions presented in [118]. First, it is assumed that the size of the input and measurement spaces are $O[n]$. Furthermore, it is assumed that that the functions $\boldsymbol{f}$ and $\boldsymbol{h}$ can be evaluated in $O[n^2]$ time, and the functions M and N can be evaluated in $O[n^3]$ time, which is the case if each element requires $O[n]$ time to compute. The size of the variance is n^2, which is implied by the size of the state space. Referencing Equations 20 and 33 and by the assumption on the time of matrix multiplication and inversion, this implies that Φ and W each require $O[n^3]$ time to evaluate. The function c_k is assumed to

be quadratic in the size of the mean and input but linear in the variance, which would imply that it takes $O[n^2]$ time to evaluate.

Throughout this analysis, the time complexity of several numerical derivatives is indicated rather than the time complexity of the analytic derivative. This is because the time to take an analytic derivative is largely dependent on the function being differentiated, while the numerical derivative can be found in predictable time using the assumptions. In many cases, using the analytic derivative may be more or less efficient than the numerical derivative, so it is up to user discretion to implement the most efficient form. The derivatives F_k, H_k, and G_k may be found numerically in $O[n^3]$. The numeric partial derivatives in Equations 67, 69, and 85-86 can each be determined in $O[n^3]$. Each of these derivatives must be performed $O[n]$ times in order to calculate the vectors $\boldsymbol{a}_k$, $\boldsymbol{b}_k$, $\boldsymbol{t}_k$, and $\boldsymbol{v}_k$, so these vectors can be computed in $O[n^4]$ time.

The complexity analysis thus far has been consistent with the results given by [118]. However, the variance derivatives calculated in Section 4.4.1 are calculated more efficiently in this work than they are in [118]. The matrices X_k, and V_k from Equations 62 and 79 may be determined analytically in $O[n^3]$ time, and each are done in lieu of computations which are $O[n^4]$ in [118]. This is because the method presented in [118] requires a partial derivative with respect to each element of P_k. Each partial derivative requires $O[n^2]$ time to evaluate, and there are n^2 elements in P_k, so in total $O[n^4]$ time is required to differentiate with respect to every element, which results in an $n^2 \times n^2$ matrix. Then, the $n^2 \times n^2$ matrix is multiplied with a vector of dimension n^2, which also requires $O[n^4]$ time. This occurs in two instances, which are given in Appendix A as Equations 177 and 181. These products are the same as what is given in Equations 62-63

and 79-80 of this work, respectively. As noted by [118], all other computations take at most $O[n^3]$ time, which is also true for this work.

Therefore, a single Bellman backup using the method presented here requires $O[n^4]$ time, which is the same order as what is presented in [118]. However, this method is strictly more efficient because the calculation of variance derivatives is performed more efficiently, while all other calculations remain the same. The advantages are even more pronounced in special situations. For example, the complexity analysis presented here did not account for situations in which measurement and input spaces may be significantly smaller than the state space, or for when the functions $\boldsymbol{f}$ and $\boldsymbol{h}$ have special structures (e.g. linearity), which may allow analytic differentiation of the vectors $\boldsymbol{a}_k$, $\boldsymbol{b}_k$, $\boldsymbol{t}_k$, and $\boldsymbol{v}_k$ in time less that $O[n^4]$. If the structure of c_k (e.g. the form used in Section 6.2.1) also allows for the efficient analytic differentiation of Q_k, R_k, J_k, U_k, $\boldsymbol{p}_k$ and $\boldsymbol{q}_k$ in time less than $O[n^4]$, then the modifications employed in this paper can enable an increase in efficiency by up to an order of magnitude. In these cases, the bottleneck for the method presented by [118] would be the calculations of the partial derivatives with respect to elements in P_k, which would still take $O[n^4]$ time to evaluate. The methods presented in this work would allow an equivalent calculation to be performed in $O[n^3]$ time, which would thus allow Bellman back-propagation to be calculated in time less than $O[n^4]$ under special circumstances. Further analysis of these special circumstances will also be the subject of future work.

A full iteration of value iteration consists of ℓ Bellman backups, so the complexity of a single iteration is $O[\ell n^4]$. The number of iterations required to converge to a locally optimal trajectory cannot be expressed in terms of ℓ or n, but convergence to

the locally optimal trajectory occurs with a second-order convergence rate, as noted by [118].

5 MODELING

To evaluate the feasibility of catching heuristics, most researchers to date have considered simple models for the ball's trajectory, either by modeling the ball's trajectory as parabolic or by including a drag force. While it is desirable to evaluate control paradigms using high fidelity models of ball and fielder dynamics (e.g. inclusion of Magnus forces), more research is necessary using simplified models to resolve several outstanding issues related to both heuristic and optimal control. For example, previous works into stochastic optimal control in the outfielder problem (including [11] and [38]) assume that the global position of the fielder may be directly measured at each time step with a high degree of accuracy and that the full global position of the ball may be directly measured with accuracy that may be state dependent. This work assumes that the only measurement that the fielder receives is the relative direction from the fielder to the ball in the fielder's local coordinate system, which measured with noise by a camera. Additionally, it does not appear that any previous work has included the uncertainty in the fielder's heading. Belousov's [11] model includes process noise in the fielder's heading, however it is measured with zero uncertainty at each time step, which effectively nullifies the uncertainty introduced by the process noise and is not feasible for practical fielders. The inclusion of uncertainty in the fielder's estimate of their heading is important because it is expected that the fielder's sense of global direction will drift as they fixate on the ball. This noise is also relevant in the implementation of heuristic controllers, which has not been evaluated in previous research. So, while it is desirable to

have a high-fidelity model of ball dynamics, there are other important factors which must be evaluated first in the parabolic model, for which the heuristics were originally designed.

Additionally, this work seeks to remove the assumption of maximum likelihood observations in predictive control. While progress is made in this regard, a maximum likelihood assumption is still used for the prediction of the time-to-impact. There also remain more outstanding issues related to input constraints which are also not resolved in this work. This work contributes to the steady progress of predictive models, but there are important issues that were not resolved here that require further study before progressing to models with higher fidelity.

5.1 BALL TRAJECTORY MODEL

The ball's motion was modeled with a deterministic parabolic trajectory with uncertain initial conditions. A deterministic trajectory was chosen to minimize the number of noise parameters for which the sensitivity analysis was performed, and the case in which ball's trajectory is deterministic given the initial conditions was complex enough to provide rich information about behaviors of each control paradigm.

The state of the ball is thus fully represented by its three-dimensional position and velocity, $\boldsymbol{x}_b = [x_b \quad y_b \quad z_b \quad \dot{x}_b \quad \dot{y}_b \quad \dot{z}_b]^T$, and the continuous time transition function is given by

$$\dot{\boldsymbol{x}}_b = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \\ 0 \\ 0 \\ g \end{bmatrix} \tag{108}$$

where $g = -9.81\ m/s^2$ is the acceleration of gravity. The solution to this differential equation is, of course, a parabola

$$\boldsymbol{p}_b[t_0, t] = \begin{bmatrix} x_b[t] \\ y_b[t] \\ z_b[t] \end{bmatrix} = \begin{bmatrix} x_b[t_0] + \dot{x}_b[t_0](t - t_0) \\ y_b[t_0] + \dot{y}_b[t_0](t - t_0) \\ z_b[t_0] + \dot{z}_b[t_0](t - t_0) + \frac{1}{2} g(t - t_0)^2 \end{bmatrix} \tag{109}$$

While this analytic form of a parabola is used to generate sample trajectories for the ball, the fielder implements first order backward Euler integration throughout this work to perform discretization of continuous time transition functions.

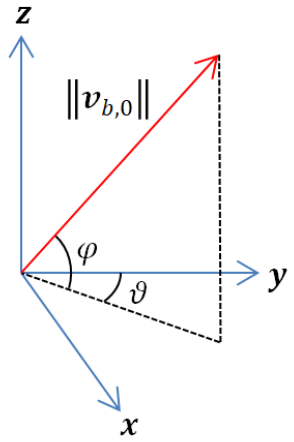
$$\boldsymbol{x}_{b,k+1} = \begin{bmatrix} x_{b,k} + \dot{x}_{b,k}\Delta t \\ y_{b,k} + \dot{y}_{b,k}\Delta t \\ z_{b,k} + \dot{z}_{b,k}\Delta t \\ \dot{x}_{b,k} \\ \dot{y}_{b,k} \\ \dot{z}_{b,k} + g\Delta t \end{bmatrix} \tag{110}$$

The length of the time step Δt was chosen to be the maximal value $\Delta t \leq 0.03\ s$ such that the time in which the ball lands, t_{impact}, is an integer multiple of Δt. First, t_{impact} is solved for in Equation 109 with $z_b = 0$ to find the time at which the ball lands. The value of t_{impact} is used to find the length of the planning horizon, ℓ.

$$\ell = \text{ceil}\left[t_{impact}/0.03\ \right] \tag{111}$$

Then, the size of the time step Δt was determined by discretizing the interval $\left[0, t_{impact}\right]$ into ℓ equally-sized segments: $\Delta t = t_{impact}/\ell$. This way, the ball will land at time step ℓ and each time step is uniform, otherwise the length of the final time step would have to be abbreviated in order to determine the state of the ball and the fielder at the time in which the ball lands. The variation in the time step size is small between trials ($< 1\%$ difference), so its effect on the numerical integration is insignificant – especially since the equations of motion for the ball are linear and the nonlinearities in the fielder's dynamics

Figure 10: Geometric description of the random parameters used to generate the ball's initial state.



are only due to a small rotation angle. The variation in the step size also has an impact of the noise model as well, since the rate at which measurements are received fluctuates slightly from trial to trial. However, because the maximal difference in the time step difference is small, this effect is insignificant. Additionally, each controller is applied to the same data sets, so whatever effects do exist are similarly experienced fairly amongst each control paradigm.

To reduce the number of free parameters, the ball is always initialized at the origin. The initial velocity of the ball is determined by randomly generating a magnitude and direction.

$$\boldsymbol{v}_{b,0} = \begin{bmatrix} \|\boldsymbol{v}_{b,0}\| \sin[\vartheta] \sin[\varphi] \\ \|\boldsymbol{v}_{b,0}\| \cos[\vartheta] \sin[\varphi] \\ \|\boldsymbol{v}_{b,0}\| \cos[\varphi] \end{bmatrix}, \tag{112}$$

$$\|\boldsymbol{v}_{b,0}\| \sim \mathcal{N}[30, 3^2], \qquad \varphi \sim \mathcal{N}[\pi/4\,, (5\pi/180)^2], \qquad \vartheta \sim \mathcal{N}[0\,, (6\pi/180)^2]$$

The variance of the initial velocity $\mathrm{var}[\boldsymbol{v}_{b,0}]$ was determined via a first order approximation of Equation 112.

5.2 FIELDER MODEL

The state of the fielder is given by its position (x_f, y_f) and velocity $(\dot{x}_f, \dot{y}_f)$ in the global reference frame, and the rotation angle θ_f which describes the orientation of the fielder reference frame with respect to the global reference frame, so that the fielder's state $\boldsymbol{x}_f$ is defined as $\boldsymbol{x}_f = [x_f \quad y_f \quad \dot{x}_f \quad \dot{y}_f \quad \theta_f]^T$. The input $\boldsymbol{u} = [u_x \quad u_y]^T$ is a specific force – a force divided by the mass of the fielder with units of acceleration – that is applied in the fielder's reference frame, and the magnitude of the input is constrained $\|\boldsymbol{u}\| \leq u_{max}$. Similar, to Belousov [11], a linear damping coefficient b is applied to the fielder's velocity, which in effect limits the maximum velocity of the fielder given that the input $\boldsymbol{u}$ is also constrained. The values of $u_{max} = 10\ N/kg$ and $b = 10/12\ s^{-1}$ that were used in this work were determined by [11] to emulate world record sprint data.

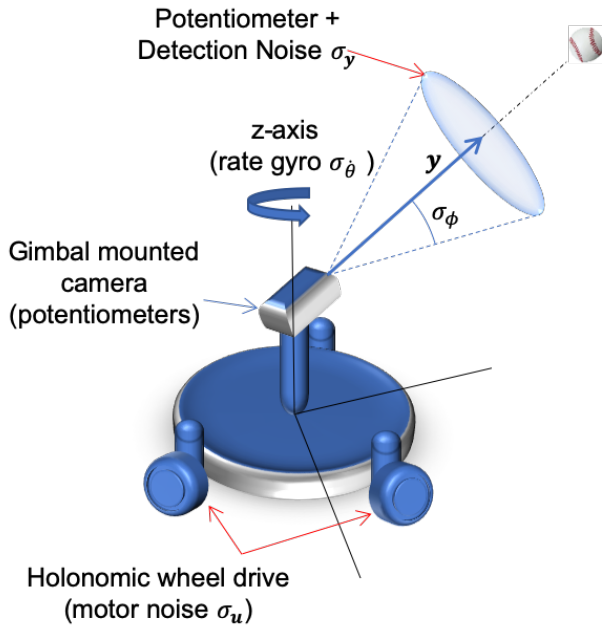


Figure 11: Artist's rendition of the fielder model.

$$\dot{\boldsymbol{x}}_f = \begin{bmatrix} \dot{x}_f \\ \dot{y}_f \\ u_x \cos[\theta_f] - u_y \sin[\theta_f] - b\dot{x}_f \\ u_x \sin[\theta_f] + u_y \cos[\theta_f] - b\dot{y}_f \\ 0 \end{bmatrix} \tag{113}$$

The continuous-time dynamics are used again in Section 6.2.1.2 to assist in cost shaping for the iLQG. However, the fielder implements first-order backward Euler integration to model the dynamics.

$$\boldsymbol{x}_{f,k+1} = \begin{bmatrix} x_{f,k} + \dot{x}_{f,k}\Delta t \\ y_{f,k} + \dot{y}_{f,k}\Delta t \\ \dot{x}_{f,k} + \left(u_{x,k} \cos[\theta_{f,k}] - u_{y,k} \sin[\theta_{f,k}] - b\dot{x}_{f,k}\right)\Delta t \\ \dot{y}_{f,k} + \left(u_{x,k} \sin[\theta_{f,k}] + u_{y,k} \cos[\theta_{f,k}] - b\dot{y}_{f,k}\right)\Delta t \\ \theta_{f,k} \end{bmatrix} \tag{114}$$

To reduce the number of free parameters, the fielder is always initialized in the same position, velocity, and orientation.

$$\boldsymbol{x}_{f,0} = \begin{bmatrix} 0 \\ 90 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{115}$$

5.3 PROCESS MODEL

The full system state $\boldsymbol{x}_k$ consists of a concatenation of $\boldsymbol{x}_{b,k}$ and $\boldsymbol{x}_{f,k}$. Additionally, additive noise $\boldsymbol{m}_k \sim \mathcal{N}[\mathbf{0}, M_k]$ is introduced to reflect motor noise and uncertainty about the angular rate:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} x_{b,k} + \dot{x}_{b,k}\Delta t \\ y_{b,k} + \dot{y}_{b,k}\Delta t \\ z_{b,k} + \dot{z}_{b,k}\Delta t \\ \dot{x}_{b,k} \\ \dot{y}_{b,k} \\ \dot{z}_{b,k} + g\Delta t \\ x_{f,k} + \dot{x}_{f,k}\Delta t \\ y_{f,k} + \dot{y}_{f,k}\Delta t \\ \dot{x}_{f,k} + \left(u_{x,k}\cos\left[\theta_{f,k}\right] - u_{y,k}\sin\left[\theta_{f,k}\right] - b\dot{x}_{f,k}\right)\Delta t \\ \dot{y}_{f,k} + \left(u_{x,k}\sin\left[\theta_{f,k}\right] + u_{y,k}\cos\left[\theta_{f,k}\right] - b\dot{y}_{f,k}\right)\Delta t \\ \theta_{f,k} \end{bmatrix} + \boldsymbol{m}_k, \tag{116}$$

$$\boldsymbol{m}_k \sim \mathcal{N}[\boldsymbol{0}, M_k],$$

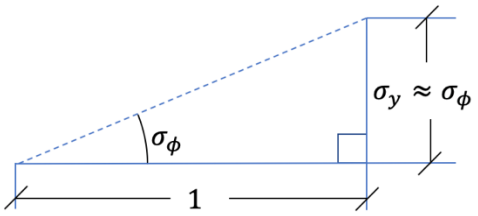
$$M_k = \mathrm{diag}\left[0, 0, 0, 0, 0, 0, 0, 0, \sigma_{\boldsymbol{u}}^2, \sigma_{\boldsymbol{u}}^2, \sigma_{\dot{\theta}}^2\right]$$

where $\sigma_{\boldsymbol{u}}^2$ reflects motor noise and $\sigma_{\dot{\theta}}^2$ reflects uncertainty in the angular rate $\dot{\theta}_{f,k}$. The values of $\sigma_{\boldsymbol{u}}^2$ and $\sigma_{\dot{\theta}}^2$ which were tested in the sensitivity analysis are given in Chapter 8.

5.4 MEASUREMENT MODEL

A camera mounted on an actuated gimbal allows tracking of the ball, and it is assumed that an independent subsystem automatically tracks the ball while keeping the ball approximately at the center of the image. The gimbal is instrumented with a sensor to measure the angle of the camera with respect to the robot's reference frame. Together, the potentiometers and the pixel coordinates of the ball in the image can be used to derive the unit vector direction from the fielder to the ball in the fielder's reference frame, which is noisy due to assumed inaccuracies of ball detection in the image and noise in the potentiometers. These two sources of error may be lumped together into one error term, which manifests itself as an angular error of the unit vector. First, let $\boldsymbol{d}_k$ be the relative distance vector from the fielder to the ball in global coordinates.

Figure 12: The image noise σ_y is approximately equal to the angular noise σ_ϕ when the angular error is small, given that the direction vector is unit length.



$$\boldsymbol{d}_k = \begin{bmatrix} d_{x,k} \\ d_{y,k} \\ d_{z,k} \end{bmatrix} = \begin{bmatrix} x_{b,k} - x_{f,k} \\ y_{b,k} - y_{f,k} \\ z_{b,k} \end{bmatrix} \tag{117}$$

The measurement $\boldsymbol{y}_k$ that the fielder receives is the direction of $\boldsymbol{d}_k$ in the fielder's reference frame, which is perturbed by the measurement noise $\boldsymbol{n}_k$

$$\boldsymbol{y}_k = \frac{1}{\|\boldsymbol{d}_k\|} \begin{bmatrix} d_{x,k} \cos[\theta_{f,k}] + d_{y,k} \sin[\theta_{f,k}] \\ -d_{x,k} \sin[\theta_{f,k}] + d_{y,k} \cos[\theta_{f,k}] \\ d_{z,k} \end{bmatrix} + \boldsymbol{n}_k, \qquad \boldsymbol{n}_k \sim \mathcal{N}[\boldsymbol{0}, N_k] \tag{118}$$

Rather than define N_k, it is more convenient to define its inverse N_k^{-1}. This is done because no information is provided about the distance from the fielder to the ball, only the direction is measured, i.e. a perturbation of $\boldsymbol{y}_k$ in the direction of $\boldsymbol{y}_k$ would change the length of $\boldsymbol{y}_k$, but not the direction. However, information is provided in the directions orthogonal to $\boldsymbol{y}_k$ – a perturbation of $\boldsymbol{y}_k$ in a direction orthogonal to $\boldsymbol{y}_k$ would change the direction of $\boldsymbol{y}_k$. To express this, let N_k^{-1} be represented as $N_k^{-1} = \mathcal{V}\Lambda\mathcal{V}^T$, where $\mathcal{V} = [\boldsymbol{y}_k \quad \boldsymbol{v}_2 \quad \boldsymbol{v}_3]^T$ and $\Lambda = \text{diag}[0, \quad \sigma_y^{-2}, \quad \sigma_y^{-2}]$. The vectors $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ are arbitrarily chosen unit vectors such that $\mathcal{V}$ is orthonormal, (which were obtained in practice via Gram-Schmidt orthonomalization [22]). If the ball is at the center of image, then the vector $\boldsymbol{y}_k$ represents the direction of the camera's optical axis in the fielder's reference frame, and the vectors $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ form a basis for the image plane expressed in the fielder's reference frame. Note that the directions of $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ need not be specific (as

long as they are orthogonal to $\boldsymbol{y}_k$) since the noise is assumed to be isotropic. If the magnitude of σ_y^2 is small, then $\sigma_y^2 \approx \sigma_\phi^2$, where σ_ϕ^2 is the variance of the angular perturbation to the direction of $\boldsymbol{y}_k$ (see Figures 11 and 12).

5.5 EXTENDED KALMAN FILTER CONSIDERATIONS

An EKF was used to perform state estimation for each controller (see Section 4.2). The ball was simulated as always starting in the same position to reduce the number of free parameters, as were the position, velocity, and orientation of the fielder. However, the initial variances of these variables were set to nonzero values to both avoid over-convergence and ensure that Γ_k is nonsingular for iLQG control. For similar reasons, small additive noise is incorporated into M_k for purposes of the EKF and iLQG stability that does not exist in the process model.

The initial variance P_0 is defined using the initial distribution with which simulated fly ball trajectories were simulated that is given in Section 5.1,

$$P_0 = \begin{bmatrix} I_{3\times3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathrm{var}[\boldsymbol{v}_{b,0}] & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{2\times2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1\mathrm{e}^{-4}I_{2\times2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 1\mathrm{e}^{-4} \end{bmatrix} \tag{119}$$

The value of M_k that was used in the EKF similarly replaces the zero-valued entries on the diagonal with small nonzero values.

$$M_k = \mathrm{diag}[1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, 1\mathrm{e}^{-6}, \sigma_{\boldsymbol{u}}^2, \sigma_{\boldsymbol{u}}^2, \sigma_{\dot{\theta}}^2] \tag{120}$$

In the measurement model, N_k^{-1} is defined as a singular matrix, therefore N_k does not exist. Therefore, the bottom definitions of the variance update and the Kalman gain are used in Equation 20 and 22 since only N_k^{-1} is required.

5.6 OBJECTIVE FUNCTION

The overall goal is to quantify the probability that the fielder will catch the ball in a particular noise configuration, where a catch is made if the fielder is within some radius ϵ of the ball at the time of impact. To express this concisely, first define χ to be

$$\chi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix} \tag{121}$$

so that

$$\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t = \left(x_{b,t} - x_{f,t}\right)^2 + \left(y_{b,t} - y_{f,t}\right)^2 \tag{122}$$

Then the continuous time reward $\mathcal{R}[\boldsymbol{x}_t, \boldsymbol{u}_t]$ is given by

$$\mathcal{R}[\boldsymbol{x}_t, \boldsymbol{u}_t] = \begin{cases} 1, & \text{if } z_{b,t} = 0 \text{ and } (\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t)^{1/2} \leq \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{123}$$

where the trial terminates at the state $z_{b,t} = 0$. If a large number of trials are simulated, then the sum of the rewards from each trial will indicate the number of successful catches, which gives an estimate of the probability of a catch for a given policy when it is divided by the number of trials. Note that the amount of effort which the fielder exerts has no influence over the total reward, nor does the agent receive any reward except at the terminal state. Therefore, the reward is considered to be sparse, which generally makes the resulting POMDP difficult to solve [68].

6 PREDICTIVE CONTROLLERS

In predictive control, the fielder can use the current state estimate of the ball to predict where the ball will land. Ideally, the fielder can then choose a running path such that the fielder will be at the landing spot when then ball arrives. For the deterministic case, there are an infinite number of controls which could be considered optimal, except

in the case where the landing spot it at the fringe of the reachable space, in which case the only solution is for the fielder to run towards the goal with maximal effort. Seemingly, however, there is an advantage to reaching the predicted landing spot as quickly as possible; the fielder would then have time to make the appropriate adjustments if there is a perturbation to the ball's trajectory or otherwise an error in the predicted landing spot. The use of a deterministic time-optimal controller has also been suggested by Gigerenzer [35]. Therefore, a deterministic time-optimal control which acts on the mean of the belief state was one method that was tested. The other predictive method is the modified iLQG method presented in Section 6.2, which is functionally the same as the one presented in [118].

In each predictive method, the predicted landing time of the ball was calculated using the current state estimate. Due how Δt was selected (see Section 5.1), the true time of impact will occur exactly at the beginning of the final time step ℓ; however, the predicted impact using the current state estimate will occur between time steps. Therefore, the length of the time step Δt for the final interval is shortened so that impact will also occur at the beginning of the final time step in the prediction.

6.1 DETERMINISTIC TIME-OPTIMAL CONTROL

The deterministic time-optimal controller has a modified objective in which it is desirable to reach and stop at the predicted landing spot in minimum time. In the case in which it is possible to reach the predicted landing spot, but not stop before the ball lands, then the objective is to arrive at the predicted landing spot as the ball lands with minimum velocity. Finally, in the case in which it is not possible to reach the predicted

landing spot before the ball lands, then it is desired to minimize the distance between the fielder and the landing spot of the ball at the time when the ball lands.

Deterministic time-optimal control was approximately achieved via a simple single-shooting method. Ideally, the time-optimal controller will be a bang-bang type controller in which the fielder is always exerting maximum effort. Analytic solutions are available for the one-dimensional case or the two-dimensional case in which box constraints are used – such that the optimal controller is essentially comprised of two independent one-dimensional solutions. However, in the case in which the magnitude of the total input is constrained within a disk, a closed form analytic solution was not found in research, so numerical methods were used. Due to linearity of the deterministic transition function, it was determined that a single-shooting-method was sufficient to quickly obtain approximately optimal results. Linearity in the fielder's motion exists because the fielder does not expect to rotate if process noise is not considered, so $\boldsymbol{u}_k$ is always applied in a fixed reference frame. However, the magnitude constraint on $\boldsymbol{u}_k$ is nonlinear. An ad hoc approach was used to achieve quick convergence to a locally optimal solution by iteratively stepping in the direction of a locally optimum sequence of inputs and then enforcing the input constraint at each time step.

First, let $\boldsymbol{u}$ to be a vector concatenation of the inputs $\boldsymbol{u}_k$ for all $k < \ell$. An initial guess $\boldsymbol{u}^{(0)}$ for the optimal input must be made; $\boldsymbol{u}^{(0)} = \mathbf{0}$ was used in this work. Then, given a guess $\boldsymbol{u}^{(i)}$ in iteration i, the predicted final state of the fielder $\boldsymbol{x}_{f,\ell}\left[\boldsymbol{u}^{(i)}\right]$ can be calculated using the fielder transition model in Equation 113. Also note that only positions and velocities are necessary to calculate, as the fielder's orientation is not controllable. A new guess for the $\boldsymbol{u}^{(i+1)}$ was then determined by:

$$\boldsymbol{u}^{(i+1)} = \boldsymbol{u}^{(i)} + \mathcal{W}^{(i)} J_{DTO}^{T} \left(J_{DTO} \mathcal{W}^{(i)} J_{DTO}^{T} + \Lambda \right)^{-1} \left(\boldsymbol{x}_{f,\ell}\left[\boldsymbol{u}^{(i)}\right] - \boldsymbol{x}_{f,goal} \right) \tag{124}$$

where Λ is a diagonal static damping matrix, $\mathcal{W}^{(i)}$ is a diagonal weighting matrix used in iteration i, $\boldsymbol{x}_{f,goal}$ is the goal position and velocity of the fielder (which is the predicted landing spot of the ball and zero velocity) and

$$J_{DTO} = \frac{\partial \boldsymbol{x}_{f,\ell}}{\partial \boldsymbol{u}} \tag{125}$$

To ensure the input constraint was not violated, it was then enforced at each time step k.

$$\boldsymbol{u}_k^{(i+1)} = \min\left[\left\| \boldsymbol{u}_k^{(i+1)} \right\|, u_{max} \right] \frac{\boldsymbol{u}_k^{(i+1)}}{\left\| \boldsymbol{u}_k^{(i+1)} \right\|} \tag{126}$$

The diagonal weighting matrix $\mathcal{W}^{(i)}$ was included to "anchor" inputs at values of k in which the input was saturated. Let $\boldsymbol{w}_k^{(i)}$ be a vector of weights which correspond to the input $\boldsymbol{u}_k^{(i)}$, such that all such vectors $\boldsymbol{w}_k^{(i)}$ form the diagonal of $\mathcal{W}^{(i)}$. The weights $\boldsymbol{w}_k^{(i)}$ are defined as:

$$\boldsymbol{w}_k^{(i)} = \begin{cases} [10^{-6}, \quad 10^{-6}]^T, & \left\| \boldsymbol{u}_k^{(i)} \right\| = u_{max} \\ [1, \quad 1]^T \quad , & otherwise \end{cases} \tag{127}$$

This effectively discounts the predicted contribution of a deviation from $\boldsymbol{u}_k^{(i)}$ if the input is saturated, since any deviation induced by Equation 124 to a saturated input will likely be reversed by the enforcement of the constraint in Equation 126. The diagonal damping matrix Λ controls the rate at which the local optimum is approached and may also be used to influence the rate of convergence with respect to each error term. Two different values of the diagonal damping matrix Λ were used depending on the objective. Thus, in a single iteration, a step is taken in the direction of the unconstrained local optimum, and then the new guess is projected back onto the constraint boundary if necessary. The net result is

thus a step along the constraint boundary towards the local minimum on the constraint boundary if projection onto the constraint boundary is necessary.

First, it was checked to see if convergence to the goal could be achieved within tolerance $\left(\left\|\boldsymbol{x}_{f,\ell}\left[\boldsymbol{u}^{(i)}\right]-\boldsymbol{x}_{f,goal}\right\|<10^{-4}\right)$ at the predicted landing time and within a maximum number of iterations (maximum iterations = 100). For this, the damping matrix $\Lambda = 10^{-4}\text{diag}[1,\ 1,\ 1,\ 1]$ was used. If convergence to the goal could be achieved within a maximum number of iterations, the planning time interval was iteratively halved using a bisection method to find the minimum time in which the goal can be reached.

If the convergence to the goal at the predicted landing time could not be achieved within a maximum number of iterations, then the fixed damping matrix Λ was modified so that convergence to the goal position is prioritized over convergence to the goal velocity: $\Lambda = \text{diag}[10^{-8},\ 10^{-8}, 10^{-3},\ 10^{-3}]$. This weighting term heavily biases minimizing the distance error compared to the velocity error, which results in behavior that causes the final distance error to be very small with respect to the velocity error. Thus, if the goal is reachable so that the distance error can be made arbitrarily small, the optimization then progresses to decrease the velocity as much as possible. If the goal is not reachable, then the planner effectively ignores the velocity error in favor of minimizing the distance error. Thus, both cases in which the goal position may or may not be reachable at time step ℓ are considered simultaneously using the same damping matrix.

An approximately optimal sequence of inputs undergoes a single abrupt transition where the input jumps from one region of the constraint boundary to another. At time steps close to the transition time, the input may not be at maximum effort. This is due to

the algorithm's inability to find a solution using maximum effort over the whole interval (except for maybe at a single time step) when the planning horizon is one-time step shorter. However, maximum effort over the whole interval is usually not required for the length of the planning horizon in which the solution was found.

6.2 iLQG CONTROL

The second predictive method that was used was the belief space variant of iLQG. However, the belief space variant of the reward function given in Equation 123 is not amenable to the iLQG method presented in Chapter 4 because there are beliefs in which the Hessian is indefinite, which violates the assumptions given in Equation 16[8]. Additionally, there are constraints on the magnitude of the input, which are not directly accounted for in the back-propagation equations. Therefore, it was necessary to shape a cost function [68] which would optimize the total expected reward by proxy.

6.2.1 COST SHAPING

Developing a sufficient cost function is complicated by the fact that the time of impact is uncertain. At each time step k, there is a probability that the ball lands (i.e. $z_{b,k} = 0$) and the trial ends, and there is a probability that the ball does not land (i.e. $z_{b,k} > 0$) and the trial continues. However, it was difficult to develop a cost function which reflected the probability of the ball landing across multiple time steps while also satisfying the constraints on the Hessians in Equation 16 and exhibiting the desired

[8] See Appendix D.

behavior[9]. Therefore, a maximum likelihood assumption was employed about the time of impact. Specifically, it was assumed that the ball lands with probability 1 at the time in which impact occurs in the nominal trajectory, which is also an assumption that is implicitly employed by other researchers as well, (e.g. [11] and [38]). Under this assumption, the probability of a catch is equal to the probability that the ball is within a disk of radius ϵ given that the ball has landed at time step ℓ, i.e. $z_{b,\ell} = 0$.

$$\Pr[Catch] = \Pr\left[(\boldsymbol{x}_\ell^T \chi^T \chi \boldsymbol{x}_\ell)^{1/2} \leq \epsilon \,\middle|\, z_{b,\ell} = 0\right] \tag{128}$$

As noted by Belousov [11], the probability that the ball lands within a disk is maximized when the expected error is zero and the variance of the error approaches 0^+, since $\boldsymbol{x}_\ell$ is Gaussian. The expected squared error is minimized under the same conditions, and thus it is commonly optimized by proxy because it is a convex quadratic function which is relatively easy to optimize.

$$\underset{\boldsymbol{x}_\ell}{\operatorname{argmax}}\left[\Pr\left[(\boldsymbol{x}_\ell^T \chi^T \chi \boldsymbol{x}_\ell)^{1/2} \leq \epsilon \,\middle|\, z_{b,\ell} = 0\right]\right] = \underset{\boldsymbol{x}_\ell}{\operatorname{argmin}}\left[\mathrm{E}\left[\boldsymbol{x}_\ell^T \chi^T \chi \boldsymbol{x}_\ell \middle| z_{b,\ell} = 0\right]\right] \tag{129}$$

Therefore, the cost function is given by

$$c_\ell[\widehat{\boldsymbol{x}}_\ell, P_\ell] = \mathrm{E}\left[\boldsymbol{x}_\ell^T \chi^T \chi \boldsymbol{x}_\ell \middle| z_{b,\ell} = 0\right] \tag{130}$$

Note that the expected value in the right side of Equation 130 is evaluated given that $z_{b,\ell} = 0$, i.e. given that the ball has landed. To simplify the evaluation of this expectation, the observation that $z_{b,\ell} = 0$ is introduced as a new kind of "measurement" which is only applied within the evaluation of the cost function $c_\ell[\widehat{\boldsymbol{x}}_\ell, P_\ell]$, since it is assumed that the ball lands at the final time step ℓ with probability 1. Therefore, the belief state at time

[9] The negated log-probability as a cost function by proxy in situations where it is desirable to maximize the probability of a Gaussian event. However, this was not well suited for the time-wise nature of this problem. See Appendix D.

step ℓ is updated using the measurement $z_{b,\ell} = 0$, which has variance that approaches 0^+ to reflect that there is no uncertainty.

$$\boldsymbol{y}'_\ell = 0 = z_{b,\ell} + n'_\ell, \qquad n'_\ell \sim \mathcal{N}[0, N'_\ell \to 0^+] \tag{131}$$

The distribution of $\boldsymbol{x}'_\ell \sim \mathcal{N}[\hat{\boldsymbol{x}}'_\ell, P'_\ell]$ is determined by applying the measurement in the above equation to the belief $\hat{\boldsymbol{x}}_\ell, P_\ell$.

$$\begin{aligned} \hat{\boldsymbol{x}}'_\ell &= \hat{\boldsymbol{x}}_\ell + K'_\ell(0 - H'_\ell \hat{\boldsymbol{x}}_\ell) \\ &= (I - K'_\ell H'_\ell)\hat{\boldsymbol{x}}_\ell \end{aligned} \tag{132}$$

$$P'_\ell = (I - K'_\ell H'_\ell) P_\ell \tag{133}$$

Thus, the belief state $\hat{\boldsymbol{x}}'_\ell, P'_\ell$ reflects that $z_{b,\ell} = 0$ is given, so the cost function at time step ℓ may simply be expressed as

$$\begin{aligned} c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell] &= \mathrm{E}[\boldsymbol{x}'^T_\ell \chi^T \chi \boldsymbol{x}'_\ell] \\ &= \hat{\boldsymbol{x}}'^T_\ell \chi^T \chi \hat{\boldsymbol{x}}'_\ell + \mathrm{tr}[\chi^T \chi P'_\ell] \end{aligned} \tag{134}$$

since the random variable $\boldsymbol{x}'_\ell$ has distribution $\mathcal{N}[\hat{\boldsymbol{x}}'_\ell, P'_\ell]$ and from the identity in Equation 37. Also, since the final time step Δt is shortened so that impact will occurs at the final time step in the prediction; it is assumed that no measurement with the camera is made at the final time step. This was done because the time interval may be quite short, so the camera may not be prepared. Additionally, this avoids possibly singular conditions when the planned final positions of the ball and the fielder are very close.

So, to summarize the methodology so far, the final cost $c_\ell[\hat{\boldsymbol{x}}_\ell, P_\ell]$ that the fielder incurs at time step ℓ is the expected squared distance between the fielder and the ball given that the ball has landed. The immediate cost $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ has not yet been defined time steps $k < \ell$. Objectively, the value $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ is 0 under the assumptions that have been presented so far. Firstly, the fielder is not at risk of missing a catch at time step

$k < \ell$ since the ball is assumed to land at time step ℓ, so it seems there is no objective reason to assign an immediate cost to a belief $\hat{\boldsymbol{x}}_k, P_k$. Secondly, the fielder has no incentive to conserve effort; instead, the fielder should exert as much effort as possible (subject to the input constraints) to minimize the total expected cost. However, in the iLQG framework, it is necessary for $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ to have a positive-definite Hessian with respect to the input in order for the algorithm to solve for a local optimal policy, so $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \neq 0$, which is further explained in the following paragraph. Furthermore, it will be demonstrated that input constraints can be addressed by assigning costs to belief states at times $k < \ell$.

6.2.1.1 Penalizing Deviations from Nominal Values

If $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = 0$, then it follows that $R_k = \mathbf{0}$. From Equations 16 and 88, it can be seen that $R_k = \mathbf{0}$ violates the constraints assumed by the iLQG algorithm. This issue was addressed by imposing the cost R_t' on the deviation of the input from the nominal value, as was mentioned in Section 4.4.3.

$$c_k'[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = \frac{1}{2}(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)^T R_k'(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) + c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \tag{135}$$

This cost essentially enforces a penalty if it is anticipated that future actions differ from the nominal input. However, this cost is also beneficial because it causes R_k to be positive definite and increasing values of R_k' incentive the inputs of the next iteration to stay close to the nominal values during replanning (see Equations 95-98 and 103). Thus, it can be used to slow the rate at which the local minimum is approached, which helps avoid overshoot and divergence. Additionally, it can be used to mitigate overestimating

the values of actions which violate the constraints. It is not believed that this method changes the argument of local optima; however, more research on this is needed.

The value of R_k' was tuned manually so that the algorithm achieved fast convergence while also having a low probability of divergence. It was found that the reliability of convergence was dependent on the length of the time horizon because small perturbations in early planning can cause large deviations later. Therefore, it was necessary to enforce a larger value of R_k' when the length of the horizon was longer because this reduced the deviations of the new trajectory from the nominal values. However, it was also determined empirically that it was desirable for each time step within a single value iteration to use the same value of R_k', or else unstable behavior resulted. The function used to determine R_k' was developed largely *ad hoc* based on these principles, and the resulting function is

$$R_k' = \frac{\operatorname{atan}\left[\frac{\ell}{5} - 4\right] - \operatorname{atan}[-4]}{2\pi - 2\operatorname{atan}[-4]} I \tag{136}$$

6.2.1.2 Cost Shaping due to Input Constraints

As noted before, input constraints are not directly accounted for in the back-propagation equations. Thus, the linear locally optimal policy given by Equation 96 does not account for inputs which violate the constraints, so the planner implicitly assumes that an agent can exploit infeasible inputs in the execution of the locally linear policy. This is especially problematic if nominal inputs are close to the constraint boundary, since the policy from iLQG will not be valid for states which are close to the nominal value.

Therefore, it was determined that intermediate belief states which maximize the fielder's reachability of highly probable landing positions were desirable[10]. From the continuous time model of the fielder in Equation 113, observe that the following is the solution for the position of the fielder $\boldsymbol{p}_f[t_0, t] = \left[x_f[t],\ y_f[t]\right]^T$ at time t when a constant input $\boldsymbol{u}$ is applied beginning at t_0:

$$\boldsymbol{p}_h[t_0, t] = \boldsymbol{p}_f[t_0] + \left(1 - e^{-b(t-t_0)}\right)\dot{\boldsymbol{p}}_f[t_0] \tag{137}$$

$$\boldsymbol{p}_p[t_0, t] = \frac{\left(b(t-t_0) - 1 + e^{-b(t-t_0)}\right)}{b^2}\boldsymbol{u} \tag{138}$$

$$\boldsymbol{p}_f[t_0, t] = \boldsymbol{p}_h[t_0, t] + \boldsymbol{p}_p[t_0, t] \tag{139}$$

where $\boldsymbol{p}_h[t_0, t]$ is the homogeneous solution, $\boldsymbol{p}_p[t_0, t]$ is the particular solution, and $\boldsymbol{p}_f[t_0, t]$ is the complete solution of the position at time t when the position $\boldsymbol{p}_f[t_0]$ and velocity $\dot{\boldsymbol{p}}_f[t_0]$ of the fielder at time t_0 are given. The particular solution $\boldsymbol{p}_p[t_0, t]$ represents the distance travelled due to the constant input $\boldsymbol{u}$. Noting that the maximum magnitude of $\boldsymbol{u}$ is the same in any direction, the reachable positions of $\boldsymbol{p}_f[t_0, t]$ form a disk with center $\boldsymbol{p}_h[t_0, t]$ and radius $\frac{\left(b(t-t_0)-1+e^{-b(t-t_0)}\right)}{b^2}u_{max}$ (see Figure 13). If $\boldsymbol{p}_h[t_0, t]$ is the same position as the expected landing spot of the ball, then the disk of reachable positions has the maximum overlap with the distribution of the error. This would allow the fielder the maximum potential to adjust to new information about the landing spot. Therefore, the fielder positions and velocities which result in the fielder arriving at the expected landing spot at the same time as the ball without applying any effort have the maximum potential to adjust to updated estimates of the landing spot.

[10] See Appendix D.

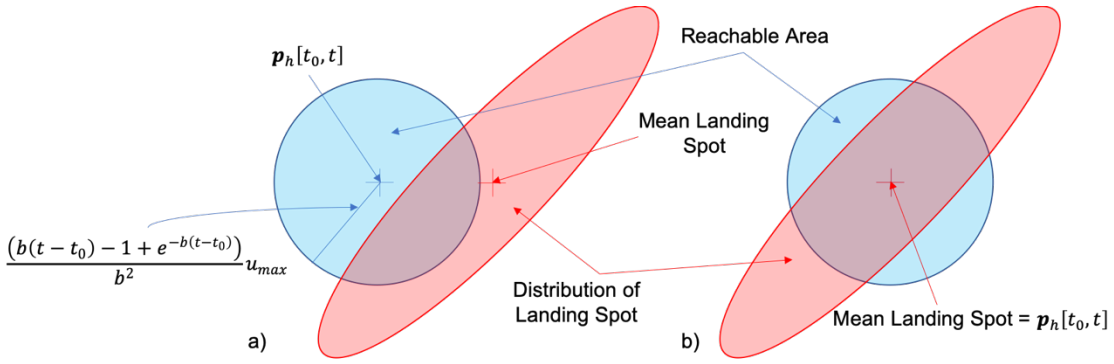


Figure 13: The area that is reachable by the fielder before impact overlaps with the highest-probability positions of the landing spot whenever the mean of the distribution of the landing spot and $\boldsymbol{p}_h[t_0, t]$ coincide.

This knowledge was used to shape the cost function to account for the input constraints. Specifically, the immediate cost $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ was weighted proportionally to the squared distance between the expected value of the homogeneous solution at the expected landing time and the expected landing spot of the ball:

$$c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] = \kappa\big(\hat{\boldsymbol{p}}_h\big[t[k], t[\ell]\big] - \hat{\boldsymbol{p}}_b\big[t[k], t[\ell]\big]\big)^T\big(\hat{\boldsymbol{p}}_h\big[t[k], t[\ell]\big] - \hat{\boldsymbol{p}}_b\big[t[k], t[\ell]\big]\big) \tag{140}$$

where $\hat{\boldsymbol{p}}_h\big[t[k], t[\ell]\big]$ returns the expected position of the fielder at time step ℓ using the state estimate at time step k and by assuming the input is zero, $\hat{\boldsymbol{p}}_b\big[t[k], t[\ell]\big]$ returns the expected position of the ball at time step ℓ using the state estimate at time k, and κ is a weighting factor. This value of $c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ was then used in Equation 135 to specify the shaped cost function $c'_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ which was used in planning. The weighting factor κ was chosen to be 2.5×10^{-3} based simply on the reasoning that the sum of the total costs of intermediate states would be weighted half as much as the terminal state given that the flight of the ball lasts 6 seconds, which is a relatively long flight among the

trajectories that were considered. However, future work should develop more rigorous justification for the value of the weighting factor κ if this method is used. The quadratic form of the cost function in Equation 140 also seems to implicate that the variance of the error term $\widehat{\boldsymbol{p}}_h\big[t[k], t[\ell]\big] - \widehat{\boldsymbol{p}}_b\big[t[k], t[\ell]\big]$ should be included in the cost function. However, the variance was omitted because its effects were not well understood at the time of this writing, but its use may be explored more in future work.

6.2.2 ITERATING UNTIL CONVERGENCE

The nominal input is updated using Equation 103. However, since Equation 103 does not consider input constraints, it is possible that the updated nominal value will be violate the input constraint. Therefore, the magnitude of the input is normalized to

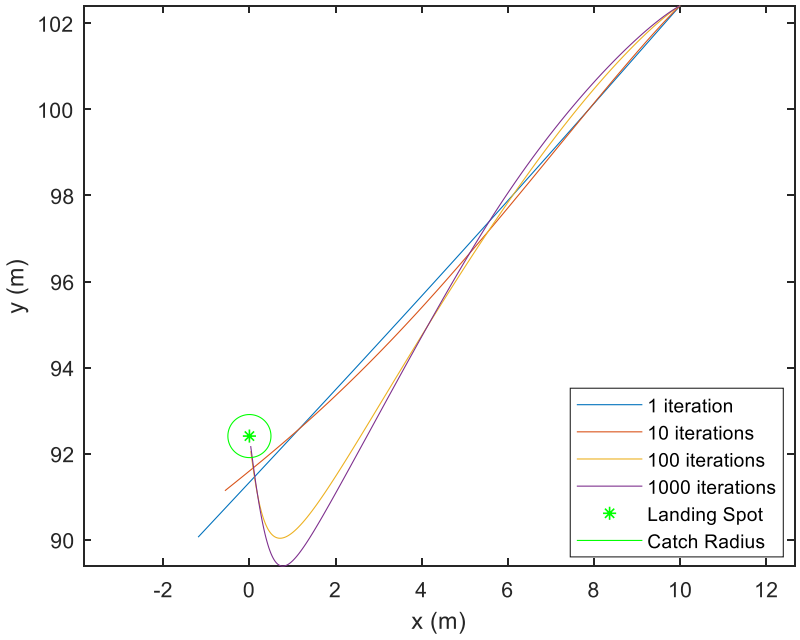


Figure 14: It can be seen that the behavior after 100 iterations of iLQG is qualitatively similar to the behavior after 1000 iterations, although the algorithm has not yet fully converged. Further iterations exhibit similar diminished returns.

maximum allowable magnitude if the constraint on the input is violated.

The iLQG algorithm is intended to be iterated until convergence (see Section 4.4.3). However, it was observed that in this model that iLQG algorithm converge slowly, which is likely due to accommodations made for the input constraint (i.e. the cost shaping of Section 6.2.1.1 and input normalization mentioned above). Therefore, an upper bound on the number of iterations that was permitted was enforced, which in this work was chosen to be 100. This did not permit full convergence of the algorithm, but it seemed to be enough to allow for good performance (see Figure 14).

7 HEURISTIC CONTROLLERS

The intent of the catching heuristics presented in Chapter 3 is to speculate how human fielders may be able to catch a fly ball despite their apparent inability to a quickly and accurately predict a ball's trajectory based on their internal model (see Section 3.1) by defining control variables which enable the fielder to decide how to act based on visual cues alone. This work differs from that premise by allowing the fielder to have access to a Bayesian state estimate to assist in the estimation and control of the controlled variables that are characteristic to the heuristic methods, as well as a one-step look ahead to predict the result of an action. Several factors motivated this approach.

First, as noted in [38], the task of making a fair comparison between heuristic and optimal approaches is complicated by the fact that each control method relies on different information. For instance, OAC requires information about the optical acceleration of the ball, whereas an iLQG controller does not need to consider optical acceleration. To determine optical acceleration for OAC, Höfer [38] uses the numerical second derivative to calculate optical acceleration from image data, since no known sensor exists which can

directly measure optical acceleration. However, numerical derivatives are sensitive to noise in the data, with the effect becoming more pronounced with higher order derivatives. Therefore, Höfer [38] proposes an alternative approach that relies on averaged velocities to mitigate these effects. In essence, Höfer [38] is seeking a more reliable way of estimating optical acceleration using readily available information, even if slightly more computation is required. However, by extension of this reasoning it is argued here that the fairest way to perform estimation of optical acceleration for the sake of comparing different controllers is to use the optimal estimate, i.e. the Bayesian estimate. Therefore, it was determined that the only way a fair comparison could be made is to allow each heuristic method to access the Bayesian estimates of the controlled variables.

Next, it is necessary to perform control so that the errors between the estimated values of the controlled variables and their set-points are immediately minimized. Only the immediate error needs to be considered since the immediate minimization of the errors is intended to lead to desirable overall behavior without the explicit consideration of the future consequences. The control methods implemented by most researchers are variants of PID control (see Section 3.5) that operate on the errors between the controlled variables and their set-points, although there remains the issue of how to properly select

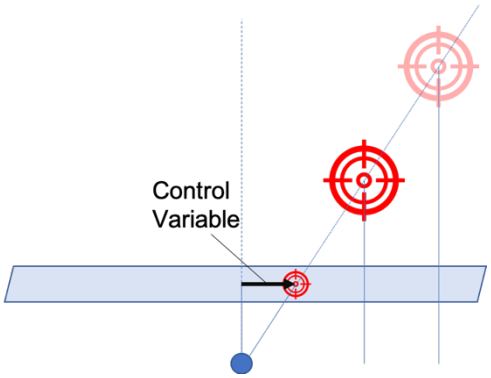


Figure 15: Consider visual servoing being performed in two dimensions, where the heuristic is to immediately align the goal with the center of the image and the control variable is the image coordinate of the goal. If the agent has access to the state estimate, then the agent may optimize the action to immediately minimize the error in the next time step, whereas control based on the error in the control variable (e.g. PID control) may be appropriate at some states but not at others.

the gains which would result in the best performance. As was noted earlier, the heuristics analyzed in this work provide desired set-points for control variables which are intended to be attained immediately. Therefore, the optimal action with respect to a given heuristic is the one that immediately minimizes the errors of the heuristic's control variables. The optimal action with respect to a given heuristic may be determined in a straightforward way by using short-term (one-step) predictions of the control variables by numerically calculating the action which results in the minimum error of the controlled variables with respect to their set-points.

While the hypothesis that catching heuristics may perform well based on limited information was not tested in this work, it should be noted that finding good policies in continuous POMDPs is a difficult problem even if the full Bayesian estimate is available. Therefore, each heuristic controller has access to an approximately Bayesian state estimate as determined by an EKF, and the controller itself is optimal in the sense that it immediately minimizes the expected error of the controlled variables. Only one-time step is considered because each heuristic approach is intended to operate using immediately available information, thus avoiding the complexity of predicting the result of a sequence of actions.

Of the catching heuristics presented in Chapter 3, two were tested in this work with the aforementioned modifications in the estimation, prediction, and control of the controlled variables. The first is based on Tresilian's method [114] (see Section 3.2), in which OAC is employed along with the requirement that the rate at which the fielder turns to track the ball is zero. However, the controller that is used to track the set-points of the control variables in this work is different from what is employed by Tresilian

because more information is available to the fielder in this work than what is assumed by Tresilian. This method will be referenced as $\dot{\delta}$-*nulling* control. The second heuristic that was tested is the generalized LOT heuristic that was presented in Section 3.4, which was based on the work of McBeath et al. [56]. Again, the fielder has access to an approximately Bayesian state estimate, which allows for different control methods than was originally proposed by McBeath et al. [56].

It will be useful in the development the heuristic controllers to define $\boldsymbol{p}_k$ as the positions of the ball and the fielder at time k:

$$\boldsymbol{p}_k = \begin{bmatrix} x_{b,k} \\ y_{b,k} \\ z_{b,k} \\ x_{f,k} \\ y_{f,k} \end{bmatrix} \tag{141}$$

The vectors $\dot{\boldsymbol{p}}_k$ and $\ddot{\boldsymbol{p}}_k$ then consist of the velocities and accelerations, respectively, of the ball and of the fielder. Under the discretization method used in this work, $\dot{\boldsymbol{p}}_k$ is not a function of the input $\boldsymbol{u}_k$ at time k. However, $\dot{\boldsymbol{p}}_{k+1}$ is a function a function of the input $\boldsymbol{u}_k$ at time k, and will be needed in the derivation of the heuristic controllers.

$$\dot{\boldsymbol{p}}_{k+1} = \begin{bmatrix} \dot{x}_{b,k+1} \\ \dot{y}_{b,k+1} \\ \dot{z}_{b,k+1} \\ \dot{x}_{f,k+1} \\ \dot{y}_{f,k+1} \end{bmatrix} = \begin{bmatrix} \dot{x}_{b,k} \\ \dot{y}_{b,k} \\ \dot{z}_{b,k} + g\Delta t \\ \dot{x}_{f,k} + \left(u_{x,k}\cos\left[\theta_{f,k}\right] - u_{y,k}\sin\left[\theta_{f,k}\right] - b\dot{x}_{f,k}\right)\Delta t \\ \dot{y}_{f,k} + \left(u_{x,k}\sin\left[\theta_{f,k}\right] + u_{y,k}\cos\left[\theta_{f,k}\right] - b\dot{y}_{f,k}\right)\Delta t \end{bmatrix} \tag{142}$$

$$\ddot{\boldsymbol{p}}_k = \begin{bmatrix} 0 \\ 0 \\ g \\ u_{x,k}\cos\left[\theta_{f,k}\right] - u_{y,k}\sin\left[\theta_{f,k}\right] - b\dot{x}_{f,k} \\ u_{x,k}\sin\left[\theta_{f,k}\right] + u_{y,k}\cos\left[\theta_{f,k}\right] - b\dot{y}_{f,k} \end{bmatrix} \tag{143}$$

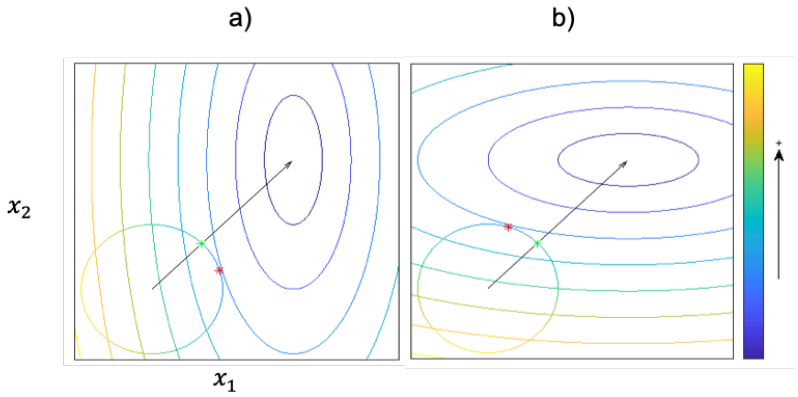


Figure 16: Constrained optimization under different weightings of parameters x_1 and x_2. In a), greater importance is assigned to the optimization of x_1, whereas in b) greater importance is assigned to the optimization of x_2. The points marked in red are the local optima on the constraint boundary, and the points marked in green are the point on the constraint boundary which is closest to the global optimum, and is invariant with respect to any weighting of x_1 and x_2.

These derivatives are utilized in the heuristic controllers in order to numerically calculate the value of the input $\boldsymbol{u}_k$ which minimizes the error of the control variables.

7.1 $\dot{\delta}$-NULLING CONTROLLER

The $\dot{\delta}$-nulling controller is based on the work of Tresilian [114] (see Section 3.2). Specifically, it was observed that if the fielder chooses a trajectory such that the optical acceleration is zero and the rate at which the fielder turns to track the ball is zero (hence $\dot{\delta}$-nulling), then the fielder will catch the ball. As a complement to OAC, basing the second constraint on the angular rate is appealing because it may be directly measured by a MEMS sensor in artificial agents.

Optimal estimates of the optical acceleration and the angular rate $\dot{\delta}$ can be obtained from the full state estimate from the EKF.

$$optical\ acceleration = \frac{d^2 \tan[\alpha_k]}{dt^2} = \frac{\partial \tan[\alpha_k]}{\partial \boldsymbol{p}_k}\ddot{\boldsymbol{p}}_k + \dot{\boldsymbol{p}}_k^T \frac{\partial \tan[\alpha_k]}{\partial \boldsymbol{p}_k \partial \boldsymbol{p}_k}\dot{\boldsymbol{p}}_k \tag{144}$$

$$\dot{\delta}_k = \frac{\partial \delta_k}{\partial \boldsymbol{p}_k}\dot{\boldsymbol{p}}_k \tag{145}$$

where

$$\tan[\alpha_k] = \frac{d_{z,k}}{\left(d_{x,k}^2 + d_{y,k}^2\right)^{1/2}} \tag{146}$$

$$\delta_k = \operatorname{atan}\left[\frac{d_{x,k}}{d_{y,k}}\right] \tag{147}$$

Note that due to the time-discretization that was used in this work, the optical acceleration at time step k is a direct function of the input at time k. However, $\dot{\delta}$ is not controllable until the following time step $k+1$, since the input needs a time step to act on the velocity $\dot{\boldsymbol{p}}_k$ before a change in $\dot{\delta}$ can be observed in the following time step (see Equations 142 and 145). Therefore, it is not possible to null the angular rate $\dot{\delta}_k$ using $\boldsymbol{u}_k$ under the method of time discretization that was employed. Instead, the input $\boldsymbol{u}_k$ will be used to null the angular rate $\dot{\delta}_{k+1}$ in the next time step:

$$\dot{\delta}_{k+1} = \frac{\partial \delta_{k+1}}{\partial \boldsymbol{p}_{k+1}} \dot{\boldsymbol{p}}_{k+1} \tag{148}$$

where $\dot{\boldsymbol{p}}_{k+1}$ (and thus $\dot{\delta}_{k+1}$) is expressible as a function of $\boldsymbol{u}_k$, as seen by Equation 142. Therefore, the controller will seek to null the optical acceleration at time step k and the angular rate $\dot{\delta}_{k+1}$ at time step $k+1$, which are the most immediate times at which the input $\boldsymbol{u}_k$ may influence each controlled variable under the discretization method used in this work. The Levenberg–Marquardt algorithm [65] with fixed damping was used to minimize the squared error, with the constraints being enforced at the end of each iteration.

$$\boldsymbol{\zeta}_{\dot{\delta}}[\boldsymbol{u}_k] = \begin{bmatrix} \dfrac{d^2 \tan[\alpha_k]}{dt^2} \\ \dot{\delta}_{k+1} \end{bmatrix} \tag{149}$$

$$\boldsymbol{u}_k^{(i+1)} = \boldsymbol{u}_k^{(i)} + \left(J_{\dot{\delta}}^T J_{\dot{\delta}} + \Lambda\right)^{-1} J_{\dot{\delta}}^T \boldsymbol{\zeta}_{\dot{\delta}}\left[\boldsymbol{u}_k^{(i)}\right] \tag{150}$$

where Λ is a static damping matrix and

$$J_{\dot{\delta}} = \left.\frac{\partial \boldsymbol{\zeta}_{\dot{\delta}}}{\partial \boldsymbol{u}_k}\right|_{\boldsymbol{u}_k = \boldsymbol{u}_k^{(i)}} \tag{151}$$

Only small damping $\Lambda = 10^{-6} I$ was needed to ensure convergence. To ensure the constraint on the input was not violated, the constraint was enforced at the end of each iteration:

$$\boldsymbol{u}_k^{(i+1)} = \min\left[\left\|\boldsymbol{u}_k^{(i+1)}\right\|, u_{max}\right] \frac{\boldsymbol{u}_k^{(i+1)}}{\left\|\boldsymbol{u}_k^{(i+1)}\right\|} \tag{152}$$

Thus, in a single iteration, a step is taken in the direction of the unconstrained local optimum, and then the new guess is projected back onto the constraint boundary if necessary. The net result is thus a step along the constraint boundary towards a local minimum on the constraint boundary. This algorithm converges to some point on the constraint boundary between the local optimum on the constraint boundary and the closest point to the unconstrained local optimum on the constraint boundary. When the value of the damping matrix Λ is large, convergence will be closer to the local optimum on the constraint boundary. In this work, the value of Λ is small, so convergence is nearer to the closest point to the unconstrained local optimum on the constraint boundary. It should be noted that the local optimum on the constraint boundary can be manipulated by changing the weighting of the controlled variables which results in different fielding behaviors; however, these effects were not explored in this work.

7.2 GENERALIZED LOT CONTROLLER

The generalized LOT controller is based of the work of McBeath et al. [56], in which an equivalent set of conditions which are amiable for control were developed in Section 3.4. As was mentioned in Section 3.4, the generalized LOT heuristic provides a general condition for a linear optical trajectory between the ball and home plate independent of the rotation of the camera's image plane, provided that the horizon is always oriented in the same direction in the image. However, the most sensible option would be for the fielder to fix their gaze on the ball, as was implemented in [120], which is the assumption that is used in this work. First, note that the direction vector from the fielder to the ball may be parameterized using the angles α and δ (see Figure 10), which may also be used to define the orientation of the camera's reference frame with respect to the global reference frame by assuming that the orientation of the horizon is horizontal in the image. The rotational transformation $R_{\alpha,k}R_{\delta,k}$ describes the sequence of rotations from the global reference frame to the camera reference frame.

$$R_{\delta,k} = \begin{bmatrix} \cos[\delta_k] & \sin[\delta_k] & 0 \\ -\sin[\delta_k] & \cos\delta_k & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{153}$$

$$R_{\alpha,k} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos[\alpha_k] & \sin[\alpha_k] \\ 0 & -\sin[\alpha_k] & \cos[\alpha_k] \end{bmatrix} \tag{154}$$

The relative position vector $\boldsymbol{d}_{b,k}$ between the fielder and the ball can thus be expressed in the camera's reference frame.

$$\boldsymbol{d}'_{b,k} = \begin{bmatrix} d'_{bx,k} \\ d'_{by,k} \\ d'_{bz,k} \end{bmatrix} = R_{\alpha,k}R_{\delta,k}\boldsymbol{d}_{b,k} \tag{155}$$

A standardized pinhole camera model is assumed to find the position of the ball $\boldsymbol{v}_{b,k}$ in image coordinates:

$$\boldsymbol{v}_{b,k} = \frac{1}{d'_{by,k}} \begin{bmatrix} d'_{bx,k} \\ d'_{bz,k} \end{bmatrix} \tag{156}$$

The position of home plate $\boldsymbol{v}_{h,k}$ in the image coordinates may be found similarly:

$$\boldsymbol{d}'_{h,k} = \begin{bmatrix} d'_{hx,k} \\ d'_{hy,k} \\ d'_{hz,k} \end{bmatrix} = R_{\alpha,k} R_{\delta,k} \begin{bmatrix} -x_{f,k} \\ -y_{f,k} \\ 0 \end{bmatrix} \tag{157}$$

$$\boldsymbol{v}_{h,k} = \frac{1}{d'_{hy,k}} \begin{bmatrix} d'_{hx,k} \\ d'_{hz,k} \end{bmatrix} \tag{158}$$

The relative distance $\boldsymbol{r}_k$ and velocity $\dot{\boldsymbol{r}}_k$ from home plate to the ball in image coordinates is thus:

$$\boldsymbol{r}_k = \boldsymbol{v}_{b,k} - \boldsymbol{v}_{h,k} \tag{159}$$

$$\dot{\boldsymbol{r}}_k = \frac{\partial \boldsymbol{r}_k}{\partial \boldsymbol{p}_k} \dot{\boldsymbol{p}}_k + \frac{\partial \boldsymbol{r}_k}{\partial \delta_k} \dot{\delta}_k + \frac{\partial \boldsymbol{r}_k}{\partial \alpha_k} \dot{\alpha}_k \tag{160}$$

and the magnitude of the angle between $\boldsymbol{r}_k$ and $\dot{\boldsymbol{r}}_k$ may then be determined from the dot product.

$$|\gamma_k| = \operatorname{acos}\left[\frac{\boldsymbol{r}_k \cdot \dot{\boldsymbol{r}}_k}{\|\boldsymbol{r}_k\| \|\dot{\boldsymbol{r}}_k\|}\right] \tag{161}$$

The generalized LOT heuristic specifies that $\gamma_k = 0$. Note that since $|\gamma_k|^2 = \gamma_k^2$, it is not necessary to determine the sign of γ_k in order to minimize the squared error with respect to the reference. Similar to the controlled variable $\dot{\delta}_k$ in the $\dot{\delta}$-nulling controller, the variable γ_k at time step k cannot be expressed as a function of the input $\boldsymbol{u}_k$ under the discretization used in this work, so it is not possible to null the angle γ_k using $\boldsymbol{u}_k$. Instead, the input $\boldsymbol{u}_k$ will be used to null the angle γ_{k+1} in the next time step:

$$\boldsymbol{r}_{k+1} = \boldsymbol{v}_{b,k+1} - \boldsymbol{v}_{h,k+1} \tag{162}$$

$$\dot{\boldsymbol{r}}_{k+1} = \frac{\partial \boldsymbol{r}_{k+1}}{\partial \boldsymbol{p}_{k+1}} \dot{\boldsymbol{p}}_{k+1} + \frac{\partial \boldsymbol{r}_{k+1}}{\partial \delta_{k+1}} \dot{\delta}_{k+1} + \frac{\partial \boldsymbol{r}_{k+1}}{\partial \alpha_{k+1}} \dot{\alpha}_{k+1} \tag{163}$$

$$|\gamma_{k+1}| = \operatorname{acos}\left[\frac{\boldsymbol{r}_{k+1} \cdot \dot{\boldsymbol{r}}_{k+1}}{\|\boldsymbol{r}_{k+1}\| \|\dot{\boldsymbol{r}}_{k+1}\|}\right] \tag{164}$$

The generalized LOT heuristic is combined with OAC in the controller implemented in this work. Therefore, the controller attempts to null the optical acceleration at time step k and the angle γ_{k+1} at time step $k+1$, which are the most immediate times at which the input $\boldsymbol{u}_k$ may influence each controlled variable. Similar to the $\dot{\delta}$-nulling controller, the Levenberg–Marquardt algorithm [65] with fixed damping was used to minimize the squared error, with the constraints being enforced at the end of each iteration.

$$\boldsymbol{\zeta}_{LOT}[\boldsymbol{u}_k] = \begin{bmatrix} \dfrac{d^2 \tan[\alpha_k]}{dt^2} \\ |\gamma_{k+1}| \end{bmatrix} \tag{165}$$

$$\boldsymbol{u}_k^{(i+1)} = \boldsymbol{u}_k^{(i)} + (J_{LOT}^T J_{LOT} + \Lambda)^{-1} J_{LOT}^T \boldsymbol{\zeta}_{LOT}\left[\boldsymbol{u}_k^{(i)}\right] \tag{166}$$

where Λ is a static damping matrix and

$$J_{LOT} = \left.\frac{\partial \boldsymbol{\zeta}_{LOT}}{\partial \boldsymbol{u}_k}\right|_{\boldsymbol{u}_k = \boldsymbol{u}_k^{(i)}} \tag{167}$$

Only small damping $\Lambda = 10^{-6} I$ was needed to ensure convergence. To ensure the constraint on the input was not violated, the constraint was enforced at the end of each iteration:

$$\boldsymbol{u}_k^{(i+1)} = \min\left[\left\|\boldsymbol{u}_k^{(i+1)}\right\|, u_{max}\right] \frac{\boldsymbol{u}_k^{(i+1)}}{\left\|\boldsymbol{u}_k^{(i+1)}\right\|} \tag{168}$$

Thus, in a single iteration, a step is taken in the direction of the unconstrained local optimum, and then the new guess is projected back onto the constraint boundary if necessary. The net result is thus a step along the constraint boundary towards the local minimum on the constraint boundary. Similar to the $\dot{\delta}$-nulling controller, this algorithm converges to some point on the constraint boundary between the local optimum on the constraint boundary and the closest point to the unconstrained local optimum on the constraint boundary. Different behaviors may result by varying the weighting of the controlled variables; however, no other weights were tested for this method either.

8 SIMULATION, RESULTS, AND DISCUSSION

A data set consisting of 500 parabolic trajectories which were reachable by the fielder before the time of impact was simulated. The distribution of the initial conditions is described in Section 5.1, and Equation 138 was used to determine if ball was reachable by the fielder. In order for a ball to be considered reachable, it was required that it was possible for fielder's position to exactly coincide with the ball's position at the time of impact, so trajectories in which the fielder could theoretically catch the ball due to the catch radius were rejected if the fielder could not center itself directly beneath the ball. Figure 17 shows the distribution of the landing spots along with the reachable area of the fielder as a function of time.

Each controller was then tested in 27 different noise configurations, which were generated by 3 different levels of noise for each of the noise parameters σ_u, $\sigma_{\dot{\theta}}$, and σ_y.

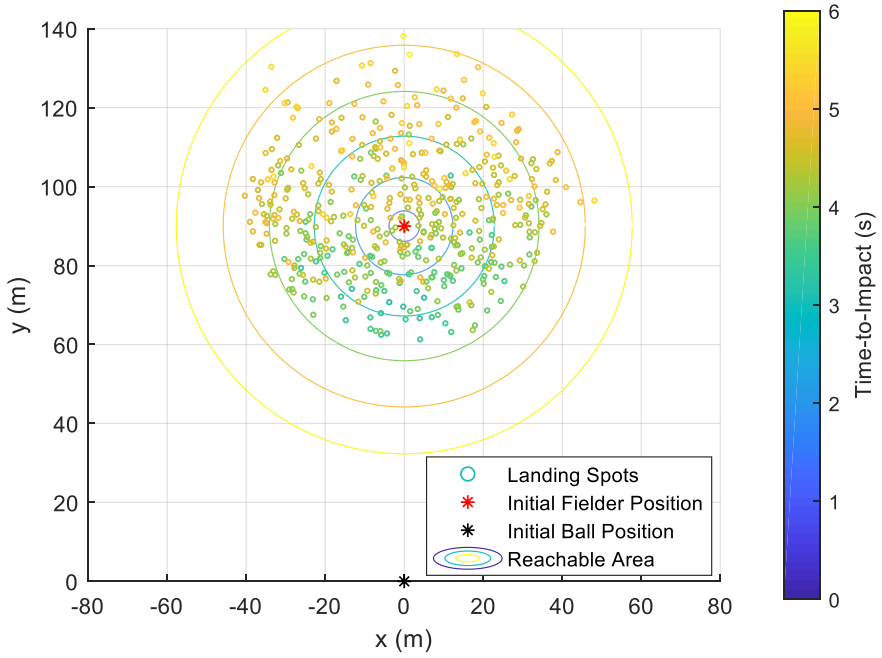


Figure 17: The distribution of the landing spots of simulated fly balls which are reachable by the fielder. The points are colored according to the length of time in which the ball is in flight, and the concentric circles indicate the boundary of the fielder's reachable area in a given amount of time.

Table 1: Settings of the noise parameters σ_u, $\sigma_{\dot{\theta}}$, and σ_y that were used in simulation.

Noise Settings			
Parameter	Small	Medium	Large
σ_u^2	$2.5e^{-3}$	$2.5e^{-2}$	$2.5e^{-1}$
$\sigma_{\dot{\theta}}^2$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$

Only the maximal noise configuration was tested for the iLQG controller. This was done because the time to generate results was significantly longer using the iLQG controller compared with the other methods; therefore, only one configuration was selected for evaluation in the interest of time. The maximal noise configuration was

selected for analysis because it produced the most statistically relevant discrepancies among the data sets generated by the other controllers; therefore, it was hypothesized that this configuration would similarly produce the most statistically relevant results for the iLQG method as well[11].

8.1 RESULTS

While each of the 500 simulated fly balls is theoretically reachable by the fielder, the running paths generated by the heuristic controllers were not efficient enough for the fielder to intercept each one in the deterministic case in which the initial states of the ball and the fielder are given and the transitions are deterministic. Unsurprisingly, the deterministic time-optimal controller was able to intercept each fly ball. The results for each controller in the deterministic configuration are given in Table 2. Since the iLQG controller was only tested in the maximal noise configuration, no results are provided for the deterministic case.

Table 2: The percentage of simulated balls which were caught by the fielder using each type of controller in the deterministic case.

Deterministic Control: Percentage of Balls Caught vs. Controller			
Controller	DTO	$\dot{\delta}$-Nulling	LOT
% of Balls Caught	100	98.4	91.2

[11] It was the opinion of the author that the reader would likely be more interested in a complete data set for a single configuration rather than partial data sets for several configurations, given that the author had only allotted time to generate results for 500 trials.

From Table 2, it can be seen that the LOT controller is already disadvantaged compared to the other controllers, which implies that the running paths generated by the LOT controller are not as efficient as the other methods.

The results of 500 trials in each of the 27 noise configurations are summarized in Tables 3-5, with 95% confidence intervals generated by assuming that the result of each trial (catch or miss) may be treated as a Bernoulli random variable. Predictably, it can be seen that the percentage of balls caught generally decreases as the value of each noise parameter increases. As the value each noise parameter is increased, the performance of controllers does not significantly degrade until somewhere between the medium and large noise settings, at which point there is a sharp decline in performance. At noise values just slightly greater than the large setting of each noise parameter was observed to render ball nearly uncatchable and were not included in this work. While the sharpest decline in the results occurs due to noise in the fielder's angular rate, this should not be interpreted to imply that the performances of the controllers are "most sensitive" to noise in the angular rate. Rather, the performances of each controller are sensitive to each noise parameter; the setting for the noise to the angular rate was simply chosen such that the performance decay was more progressed than in the other noise settings, which was not known *a priori*.

Table 3: Percentage of simulated balls caught by the fielder when using deterministic time-optimal control under each noise configuration.

Deterministic Time Optimal: Percentage of Balls Caught [95% Confidence Interval]									
σ_u^2 \ σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
$2.5e^{-3}$	97.2 95.8, 98.6	97.2 95.8, 98.6	87.0 84.0, 89.9	97.0 95.5, 98.5	97.0 95.5, 98.5	86.6 83.6, 89.6	39.2 34.9, 43.5	42.8 38.5, 47.1	35.6 31.4, 39.8
$2.5e^{-2}$	95.4 93.6, 97.2	96.4 94.8, 98.0	87.2 84.3, 90.1	96.2 94.5, 97.9	95.4 93.6, 97.2	86.6 83.6, 89.6	34.6 30.4, 38.8	34.0 29.9, 38.1	30.0 26.0, 34.0
$2.5e^{-1}$	87.4 84.5, 90.3	86.4 83.4, 89.4	65.4 61.2, 69.6	85.8 82.7, 88.9	85.4 82.3, 88.5	66.0 61.8, 70.1	34.2 30.0, 38.4	34.6 30.4, 38.8	26.8 22.9, 30.7
σ_θ^2	$1.0e^{-4}$			$1.0e^{-3}$			$1.0e^{-2}$		

Table 4: Percentage of simulated balls caught by the fielder when using $\dot{\delta}$-Nulling control under each noise configuration.

$\dot{\delta}$-Nulling: Percentage of Balls Caught [95% Confidence Interval]									
σ_u^2 \ σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
$2.5e^{-3}$	95.6 93.8, 97.4	96.6 95.0, 98.2	81.4 78.0, 84.4	94.4 92.4, 96.4	94.8 92.9, 96.7	78.2 74.6, 81.8	29.4 25.4, 33.4	29.8 25.8, 33.8	21.0 17.4, 24.6
$2.5e^{-2}$	93.6 91.5, 95.7	93.6 91.5, 95.7	74.8 71.0, 78.6	92.6 90.3, 94.9	92.4 90.1, 94.7	70.2 66.2, 74.2	23.6 19.9, 27.3	23.6 19.9, 27.3	15.8 12.6, 19.0
$2.5e^{-1}$	82.2 78.8, 85.6	81.4 78.0, 84.8	51.4 47.0, 55.8	80.0 76.5, 83.5	80.2 76.7, 83.7	48.6 44.2, 53.0	18.0 14.6, 21.4	17.0 13.7, 20.3	10.8 8.1, 13.5
σ_θ^2	$1.0e^{-4}$			$1.0e^{-3}$			$1.0e^{-2}$		

Table 5: Percentage of simulated balls caught by the fielder when using LOT control under each noise configuration.

LOT: Percentage of Balls Caught [95% Confidence Interval]									
σ_u^2 \ σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
$2.5e^{-3}$	89.6 86.9, 92.3	90.0 87.4, 92.6	79.6 76.1, 83.1	87.6 84.7, 92.3	87.6 84.7, 90.5	74.8 71.0, 78.6	23.4 19.7, 27.1	23.6 19.9, 27.3	12.6 9.7, 15.5
$2.5e^{-2}$	87.0 84.1, 89.9	86.4 83.4, 89.4	70.2 66.2, 74.2	85.4 82.3, 88.5	85.0 81.9, 88.1	66.2 62.1, 70.3	16.6 13.3, 19.9	16.4 13.2, 19.6	11.4 8.6, 14.2
$2.5e^{-1}$	69.0 64.9, 73.1	68.6 64.5, 72.7	42.8 38.5, 47.1	66.8 62.7, 70.9	66.0 61.8, 70.2	38.8 34.5, 43.1	10.0 7.4, 12.6	10.0 7.4, 12.6	5.8 3.8, 7.8
σ_θ^2	$1.0e^{-4}$			$1.0e^{-3}$			$1.0e^{-2}$		

The relative sensitivities of each controller with respect to the noise parameters may be analyzed by directly comparing the results generated by each controller. Tables 6-7 provide direct comparisons between each control method presented in Tables 3-5, with 95% confidence intervals generated by assuming that the result of each trial (catch or miss) may be treated as a Bernoulli random variable. From Table 6, it can be seen that the deterministic time-optimal controller significantly dominates the $\dot{\delta}$-nulling in nearly every noise configuration. It can be seen that the significance of the dominance is amplified as the noise is increased, indicating the performance of the $\dot{\delta}$-nulling controller degrades more rapidly in the presence of noise. Similarly, Table 7 demonstrates the dominance of the $\dot{\delta}$-nulling controller versus the LOT controller in every noise configuration, which transitively also indicates the dominance of the deterministic time-optimal controller over LOT.

Table 6: Each cell indicates the percentage-point differential between deterministic time-optimal and $\dot{\delta}$-Nulling control in each noise configuration. Positive values indicate a greater percentage of balls are caught using deterministic time-optimal control, and the bottom values indicate the 95% confidence interval.

DTO vs. $\dot{\delta}$-Nulling: Percent Difference [95% Confidence Interval]									
σ_u^2 \ σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
$2.5e^{-3}$	1.6 -0.7, 3.9	0.6 -1.5, 2.7	5.6 1.1, 10.1	2.6 0.1, 5.1	2.2 -0.3, 4.7	8.4 3.7, 13.1	9.8 3.9, 15.7	13.0 7.1, 18.9	14.6 9.1, 20.1
$2.5e^{-2}$	1.8 -1.0, 4.6	2.8 0.1, 5.5	12.4 7.6, 17.2	3.6 0.8, 6.4	3.0 0.0, 6.0	16.4 11.4, 21.4	11.0 5.4, 16.6	10.4 4.8, 16.0	14.2 9.1, 19.3
$2.5e^{-1}$	5.2 0.8, 9.6	5.0 0.5, 9.5	14.0 8.0, 20.0	5.8 1.1, 10.5	5.2 0.5, 9.9	17.4 11.4, 23.4	16.2 10.8, 21.6	17.6 12.3, 22.9	16.0 11.3, 20.7
σ_θ^2	$1.0e^{-4}$			$1.0e^{-3}$			$1.0e^{-2}$		

Table 7: Each cell indicates the percentage-point differential between $\dot{\delta}$-Nulling and LOT control in each noise configuration. Positive values indicate a greater percentage of balls are caught using $\dot{\delta}$-Nulling control, and the bottom values indicate the 95% confidence interval.

$\dot{\delta}$-Nulling vs. LOT: Percent Difference [95% Confidence Interval]									
σ_u^2 \ σ_y^2	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$	$1.0e^{-4}$	$1.0e^{-3}$	$1.0e^{-2}$
$2.5e^{-3}$	6.0 2.8, 9.2	6.6 3.5, 9.7	1.8 -3.1,6.7	6.8 3.3, 10.3	7.2 3.7, 10.7	3.4 -1.9, 8.7	6.0 0.5, 11.5	6.2 0.7, 11.7	8.4 3.8, 13.0
$2.5e^{-2}$	6.6 3.0, 10.2	7.2 3.5, 10.9	4.6 -0.9, 10.1	7.2 3.3, 11.1	7.4 10.9, 11.3	4.0 -1.8, 9.8	7.0 2.1, 11.9	7.2 2.3, 12.1	4.4 0.2, 8.6
$2.5e^{-1}$	13.2 7.9, 18.5	12.8 7.5, 18.1	8.6 2.4, 14.8	13.2 7.8, 18.6	14.2 8.8, 19.6	9.8 3.7, 15.9	8.0 3.7, 12.3	7.0 2.8, 11.2	5.0 1.6, 8.4
σ_θ^2	$1.0e^{-4}$			$1.0e^{-3}$			$1.0e^{-2}$		

The comparison of the iLQG controller to the other controllers in the maximal noise configuration is provided in Table 8. It can be seen that at this noise setting, iLQG outperforms the heuristic controllers by a wide margin, although its performance over the deterministic time-optimal controller was not significant. It is also important to note that the performance of the iLQG controller relative to the other controllers at other noise configurations – specifically with respect to the deterministic time-optimal controller – cannot be definitively inferred based on the singular result provided by the maximal noise configuration. Nevertheless, it seems that, contrary to observations made by [11] when studying similar heuristics (e.g. OAC and LOT), catching heuristics do not seem to generate optimal policies, despite the fact that that approximately optimal trajectories may appear to satisfy the heuristics. However, this does not imply that the catching heuristics do not generate good policies, especially if computation time is taken into account. Instead, the percentage of balls that can be caught in real time would be the most appropriate criterion by which each control method should be judged. Table 9 shows the computation time per time step of each approach, which includes the time required to perform state estimation. The comparison is carried out in Matlab on a PC with two 3.40GHz 8-Core Intel Xeon CPUs and 32GB RAM running Windows 10. Note that the iLQG controller and the deterministic time-optimal controller each have a planning time which is a function of the horizon. In the worst case, the computation time for the iLQG controller would effectively rule out the application of the iLQG controller in real time[12], while the other methods may be applied in real time.

[12] On a faster system, with more efficient coding, it is likely that iLQG could be computed in real time. However, it must also be considered that a standalone fielder would be a mobile platform.

The version of iLQG with reward shaping described in this paper does not provide a significant advantage over the deterministic time-optimal controller, especially if the computational cost of the iLQG controller with respect to the deterministic time-optimal controller is taken into account. Therefore, it is likely necessary to improve on the assumptions used in this work in order for iLQG to be considered worthwhile compared to simply employing deterministic time-optimal control when computing large data sets.

Belousov et al. [11] observed that fielders running in approximately optimal trajectories may appear to satisfy the heuristics. However, in this work, it was seen that running paths which were generated by heuristic controllers under ideal conditions resulted in significantly degraded performance. To further explore this idea, a qualitative study of the running paths generated by each control method was performed.

Table 8: First Row: Percentage of simulated balls which are caught using each controller in the maximal noise configuration. Second Row: Percent-point difference between each controller and iLQG control, where positive values indicate more balls were caught using iLQG control. The values on the bottom of each cell indicate the 95% confidence intervals.

Maximal Noise Configuration: $\sigma_u^2 = 2.5\text{e}^{-1},\ \sigma_{\dot{\theta}}^2 = 1.0\text{e}^{-2},\ \sigma_y^2 = 1.0\text{e}^{-2}$				
Controller	iLQG	DTO	$\dot{\delta}$-Nulling	LOT
% Caught 95% Confidence Interval	28.8 24.8, 32.8	26.8 22.9, 30.7	10.8 8.1, 13.5	5.8 3.8, 7.8
% Less than iLQG 95% Confidence Interval	–	2.0 -3.6, 7.6	18.0 13.2, 22.8	23.0 18.5, 27.5

Table 9: First Row: Run-time per iteration as a function of the length of the planning horizon ℓ. Second Row: Maximum run-time assuming the maximum length of the planning horizon is $\ell = 200$ (6 seconds).

Run-Time per Time-Step (seconds)				
Controller	iLQG	DTO	$\dot{\delta}$-Nulling	LOT
Run-time per time-step	0.27ℓ	$1.2e^{-4}\ell$	$7.1e^{-3}$	$8.4e^{-3}$
Maximum run-time per time-step	54.4	0.025	$7.1e^{-3}$	$8.4e^{-3}$

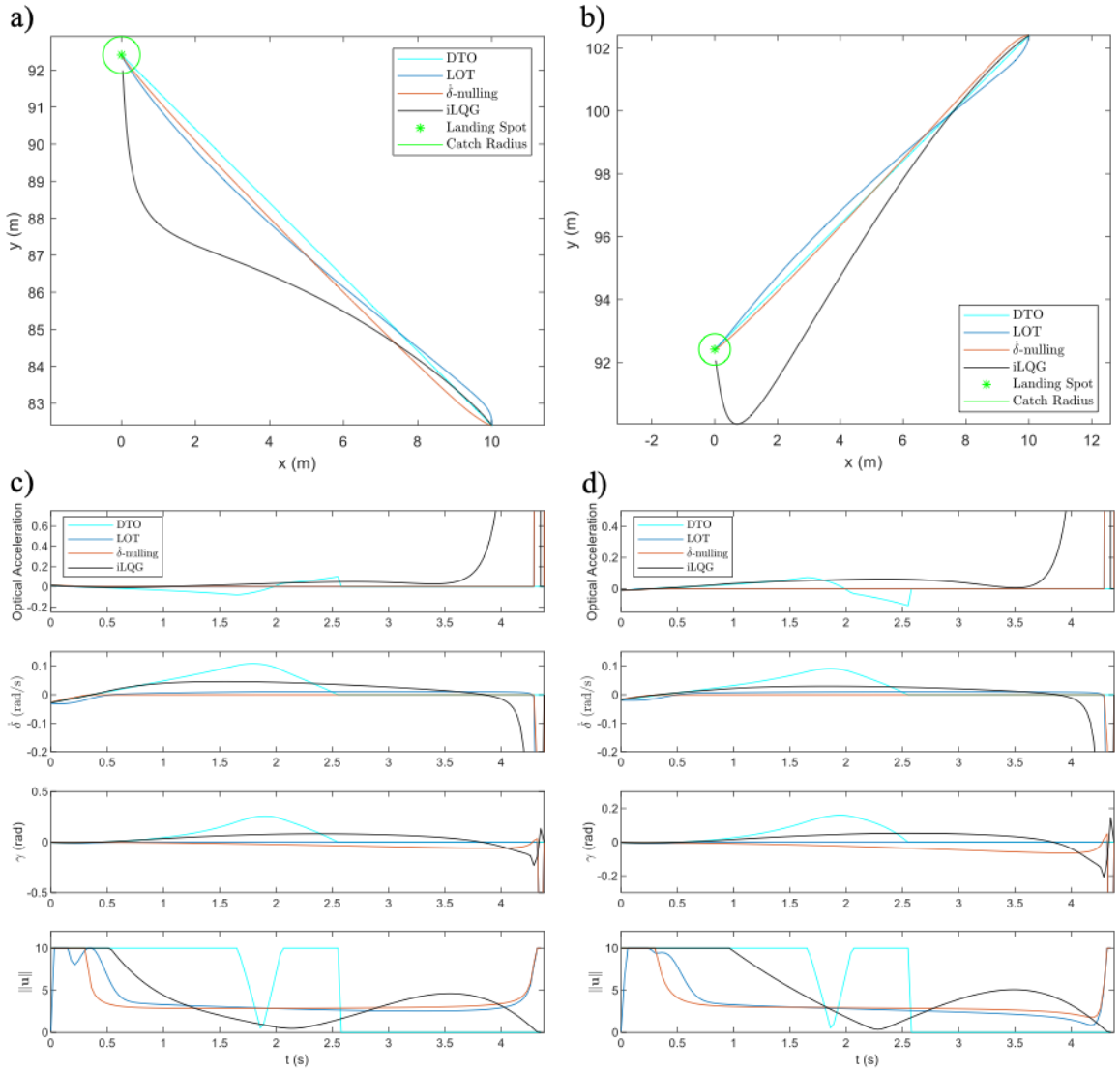


Figure 18: The figures in a) and b) show the initial planned running paths generated by each controller under different initial positions of the fielder. The figures in c) and d) provide the values of the controlled variables and input magnitude that are experienced by each path provided in a) and b), respectively.

Figure 18 shows nominal trajectories generated by iLQG given an initial state estimate in which the fielder is at rest, as well as trajectories that would be generated by the other methods under the same initial conditions. It can be seen that the nominal trajectories of iLQG have the fielder move in-line with the trajectory of the ball before moving towards the predicted landing spot while moving outwardly from home plate, whereas the other methods employ a more direct path towards the predicted landing spot.

This behavior may be qualitatively interpreted in a couple different ways. Superficially, it may seem that its best for the fielder to run in the same direction as the ball in the time interval close to impact since the fielder is uncertain about the time at which ball will land, therefore the fielder should move with the ball in case the ball impacts at moment during the time interval. Note, however, that the maximum likelihood estimate of the time-to-impact is used in planning, so that the time-to-impact is treated as deterministic. Therefore, there would be no benefit in this approach if the time-to-impact is treated as a known deterministic value. Another hypothesis which may intuitively explain this behavior is that moving as close to the ball as possible maximizes the observability of the ball, but this must also be balanced with the fielder's need to reach the predicted landing spot at the correct time. Additionally, there may also be a benefit for the fielder to move into the plane of the ball's motion, as this would maximize the observability of the direction of the ball's travel. Thus, the fielder would only have to worry about being at the correct radial distance in order to catch the ball successfully.

Note that these running trajectories generated by the iLQG controller differ significantly from the approximately optimal running paths generated in [11], which were

generated assuming maximum likelihood observations that resulted in more direct running paths similar to the heuristic controllers. The running trajectories generated by the iLQG controller also differ from observed human behavior, in which the fielder moves in line with plane of the ball's motion but with preference for approaching the ball

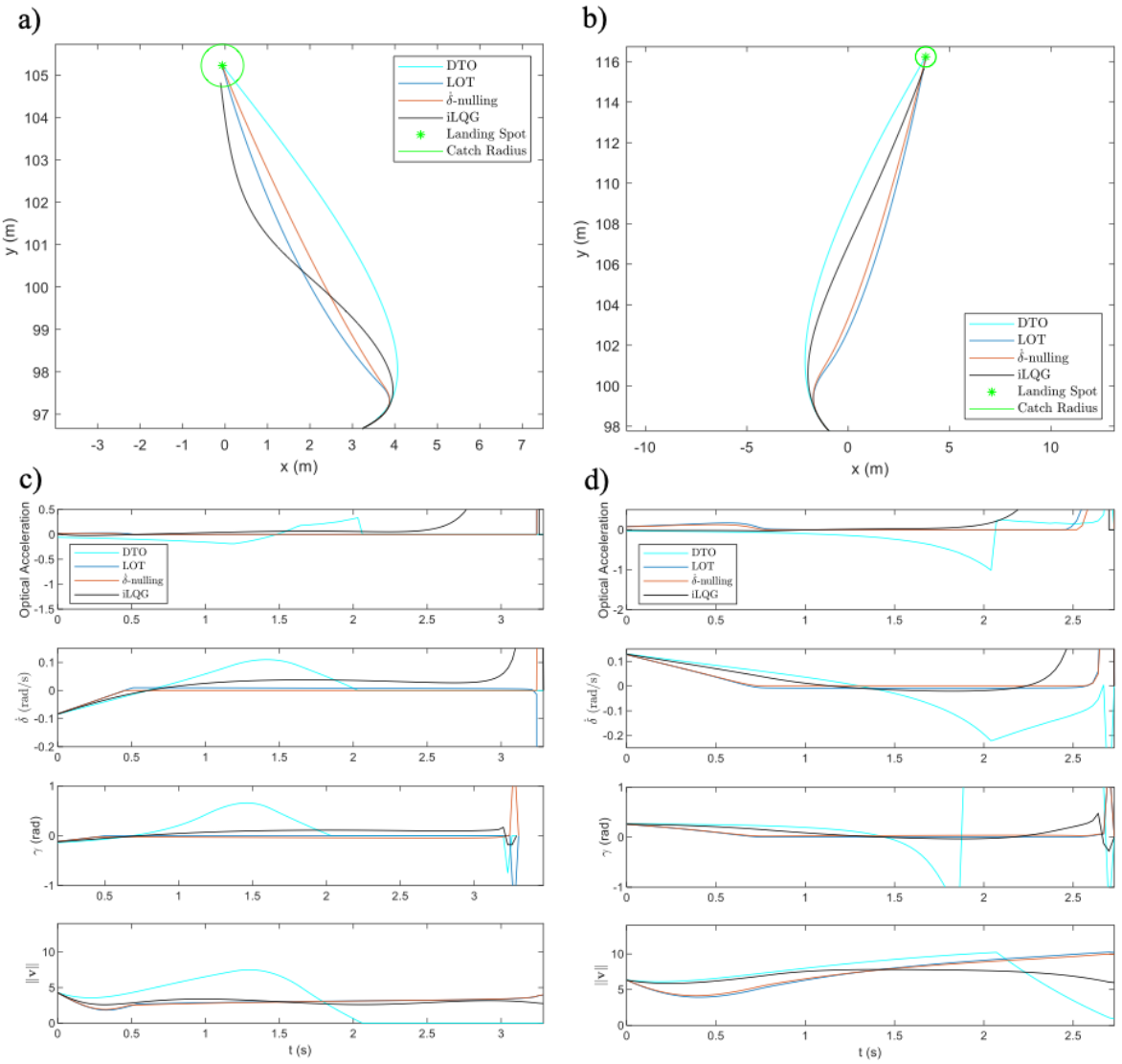


Figure 19: The figures in a) and b) show the planned running paths generated by each controller for the remainder of the trial after the first 2.25 seconds were controlled using LOT in the maximal noise configuration. The figures in c) and d) provide the values of the controlled variables and the magnitude of the fielder's velocity that are experienced by each path provided in a) and b), respectively.

while moving towards home plate rather than away from it [41]. However, the dynamics of a human fielder differ significantly from the model assumed in this work, as humans cannot accelerate equally well in any direction in the ground plane but rather are restricted by their heading. Additionally, the range of motion a human's neck also influences their trajectory so that they can maintain visual contact with the ball. Furthermore, there are other considerations that arise when a fly ball is considered within the broader context of the game of baseball, in which the fielder may more easily make a play if they catch the ball while moving towards home plate and the fielder is also incentivized for keeping the ball in front of them in general.

Also note that the fielder's nominal trajectories generated by iLQG did not terminate at the predicted landing spot. It is unknown if convergence can be achieved with more iterations, although full convergence was not seen even after 10,000 iterations, compared to the 100-iteration limit used in this work. However, it was observed that convergence to the predicted landing spot would occur under short planning horizons in which the uncertainty in the predicted landing spot is reduced, which enables the fielder to make adjustments in the final moments to get closer to the ball.

In Figures 18 c) and d), the values of the controlled variables used in $\dot{\delta}$-nulling control and LOT are shown for each running path in Figures 18 a) and b), respectively, as well as the magnitude of the input. It can be seen that the heuristic controllers are able to nullify their respective controlled variables whenever the input is not saturated. The nominal trajectory of iLQG seems to track the set-points of the controlled variables to some degree, despite having a drastically different trajectory.

In Figure 19, the fielder had been tracking the ball using LOT in the maximal noise configuration for the initial 2.25 seconds of the ball's flight. Using the state estimate at 2.25 seconds, nominal trajectories were generated by iLQG, as well as trajectories that would be generated by the other methods under the same initial conditions if maximum likelihood measurements were observed. In Figure 19 a), it is

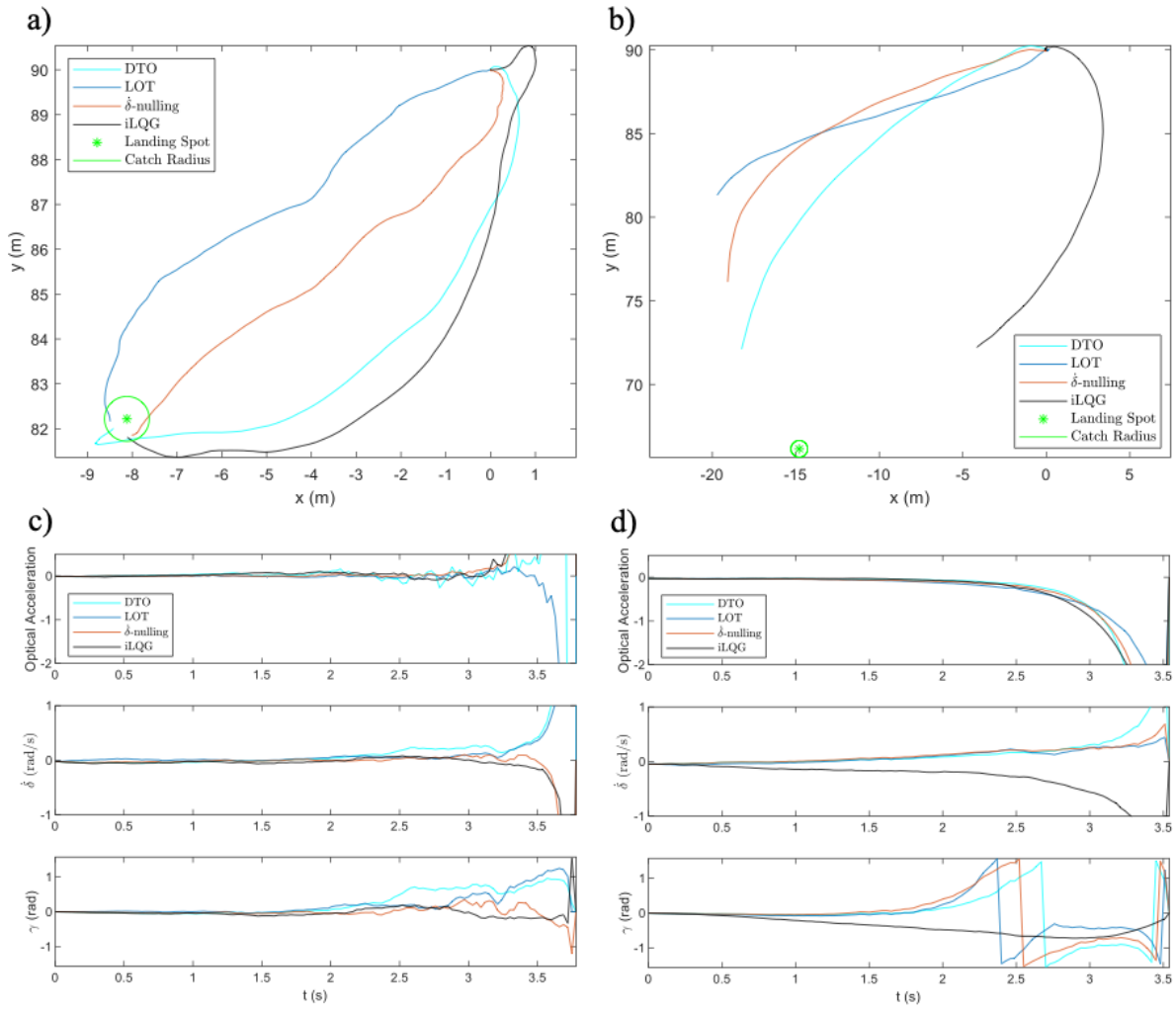


Figure 20: The figures in a) and b) show the running paths that were executed by the fielder for selected trials which resulted in a) catches and b) misses when using each controller. The figures in c) and d) provide the values of the controlled variables that are experienced by each path provided in a) and b), respectively.

possible for the fielder to reach the predicted landing spot and wait for the ball to land, whereas in Figure 19 b) the fielder must catch the ball in motion, which can be seen from the norms of the velocities of the deterministic time-optimal controllers in Figures 19 c) and d), respectively. Again, it can be seen in Figures 19 a) and b) that the nominal trajectory from iLQG tracks the set-points of the controlled variables reasonably well, despite being generated by very different running paths compared to those generated by the heuristic methods.

These results are similar to the observations made [11], in which approximately optimal running paths also appeared to track the expected set-points of the control variables in OAC and LOT. However, the running paths generated by iLQG in this work are drastically different from the running paths generated by the heuristic controllers and the controller presented in [11]. This seems to indicate that there are a wide variety of successful running paths in which the controlled variables are tracked "reasonably" well, thus it may be difficult reject a hypothesis that actual human fielding behaviors are generated by a particular catching heuristic, i.e. there exists a causal relationship. This also seems to give further credence to the hypothesis that the satisfaction of the heuristics may be a geometric consequence of the implementation of another control strategy, as postulated in [11], since the running paths presented in [11] and in this work are quite disparate yet qualitatively seem to satisfy the heuristics.

This is illustrated again in Figure 20, in which Figures 20 a) and b) show a full simulated trial in the maximal noise configuration, where the trial in Figure 20 a) resulted in a catch by each controller, and the trial in Figure 20 b) resulted in a miss by each controller. It can be seen in Figures 20 c) and d) that each method qualitatively seems to

track each of the controlled variables similarly, despite the fact that the running paths generated by each method are dissimilar aside from the fact that they terminate in the same result. This again highlights the difficulty in rejecting the hypothesis of a causal relationship when evaluating human data and lends further credence to the hypothesis that the satisfaction of the heuristics may be a geometric consequence of the implementation of another control strategy

8.2 DISCUSSION

Cost shaping and the maximum likelihood assumption on the time-to-impact was employed as a means to make the outfielder problem accessible to the iLQG algorithm. So, while iLQG converges to a locally optimal solution, this solution by proxy is only valid if the shaped cost function leads to the same behavior that would result from optimizing with respect to the original reward function (i.e. Equation 123). The error in this approximation has not been studied in this work, so the validity of the approximate solutions determined in this work is debatable. Future work in iLQG should quantify the expected error induced by these assumptions or explore more precise means of handling input constraints and uncertainty in the time-to-impact.

While one of the stated goals of this work was to statistically quantify an upper bound on the performance of the heuristic methods, there is a notable imprecision in the definition of the heuristics which makes this difficult to unambiguously achieve. Specifically, whenever the input which nullifies the control variables is not contained within the input constraint, an input on the constraint boundary which immediately minimizes the error with respect to the controlled variables is sought. However, the input on the constraint boundary which minimizes the error with respect to the control

variables is subject to the weighting of the control variables, with different weightings leading to different fielding behaviors (see Figure 16). Therefore, in order to truly find the maximum expected performance that can be achieved by *immediately* minimizing the errors of the controlled variables, the weighting of the controlled variables should be optimized. However, the time-cost of completing this task seemed to outweigh the immediate reward, so this was relinquished to future work. Alternatively, it may also be possible to quantify the maximum expected performance under *any* weighting of the controlled variables by allowing multistep prediction in the heuristic methods. However, this was not explored in this work since it seemed to convolute the intent of the heuristic methods but may be an approach employed in future work.

Regardless, it could be seen that in moderate and low noise configurations, the heuristic controllers demonstrated successful catching behavior, specifically the $\dot{\delta}$-nulling controller. While the performance of the LOT controller was dominated by every other controller that was considered, it should be noted that LOT may benefit if additional information was included, specifically background image data. It was noted in [120] that human fielding behavior changed in response to variations of the flow in the background image data, which the authors attribute to the use of the LOT heuristic. While it is possible that variations in the flow of the background image data, it is also possible that background image data directly assists in the application of a LOT-type heuristic, so that the background image flows linearly as the fielder remains fixed on the ball. However, more research on this is needed.

However, for many practical mobile platforms, the deterministic time-optimal controller presented in Section 6.1 would provide the best results, while also being

efficient enough to run in real-time. Therefore, the full benefit of heuristic controllers is only realized aboard agents in which it is impractical to obtain a full state estimate.

As was noted in the previous section, rejecting a causal relationship between human fielding behavior and the aforementioned heuristics based on observed human fielding data alone is likely very difficult due to the fact that the control variables are not very sensitive to the successful trajectories of the fielder. Therefore, there is a need for experiments which can more precisely isolate each control variable.

Perhaps the greatest insights from the study of the outfielder problem can be derived from the human reasoning that was employed in order to generate the various fielding heuristics. This may be indicative of a more general problem-solving strategy that is employed by humans which may be possible to imitate on a rudimentary level by an autonomous artificial agent. For instance, each fielding heuristic studied in this work may be reduced to a pair of control variables that are intended to be invariant throughout the flight of the ball, and satisfaction of the heuristics are guaranteed to result in successful fielding behavior. Guided by this principle, a new invariant was developed in Section 3.4, albeit the new heuristic relied heavily on the LOT heuristic developed in [56]. However, autonomously searching for invariants which result in globally desirable behavior may be a strategy which is realizable with further study.

Research into the outfielder problem has demonstrated how human reasoning may be applied to develop simple rules to guide complex decision-making. Specifically, these rules may be applicable to allow a human fielder to quickly form high-quality decisions. While the researchers developed the heuristics as a product of deliberate effort, it is also possible that human fielders similarly develop such rules subconsciously as they learn to

catch fly balls. In either case, the generated rules seek to reliably catch fly balls, while also considering the resource constraints of a human fielder.

9 CONCLUSIONS AND DIRECTION OF FUTURE WORK

This research explored methodologies which enable decision makers to form good decisions for continuous POMDPs. The methodologies that were explored included heuristic approaches, in which reasoning is applied offline to form simple rules which guide the decision-making process, and a belief space variant of iLQG [118], which is a trajectory optimization method that exploits the dynamics model online. Conventional model-free reinforcement learning methods, including model-free methods which are applied to simulated models, were not studied in this work. Rather, it is proposed that the autonomous formation of heuristics is a method by which artificial agents may actively limit their resource demands by exploiting structure within the environment and exploiting a coupling between their sensing and actuation for fast decision-making, similar to how it is hypothesized that humans perform decision-making [35]. This may be used in conjunction with, or as alternative to, conventional model-free reinforcement learning methods. However, this methodology requires further research before it is applicable to practical problems.

The contributions of this work which may have the most immediate impact are the novel modifications to a belief space variant of iLQG [118] which reduced the time complexity of computing certain matrix derivatives from $O[n^4]$ to $O[n^3]$ by employing directional derivatives. Under special circumstances, such as those encountered in this work, the calculation of these derivatives forms the computational bottleneck of the algorithm, so the efficiency of the algorithm is greatly improved by the use of directional

derivatives. In addition to the application of the modified belief space variant of iLQG as a standalone planner, it may also be employed within the framework of a sample-based path planner (e.g. [4][19]).

This work applied modified belief iLQG algorithm to a target interception problem, specifically the outfielder problem. It was noted several limitations exist which impede the direct application of the iLQG algorithm, such as the uncertainty about the time-to-impact and input constraints. Similar issues would also arise in other target interception problems, such as missile defense [126]. In this work, these limitations were circumvented through the use of cost shaping. However, in future work it would be constructive to handle these problems using a more direct method (e.g. by considering the probability that the ball will land across multiple time-steps). While iLQG was made even more efficient by the methods developed in this work, it remains impractical to implement in real-time on a practical mobile platform due to its long running time (see Table 9). However, the deterministic time-optimal controller (see Section 6.1), which solves for a minimum time path from the mean of fielder's position estimate to the predicted landing spot of the ball, is efficient enough to be implemented in real-time and also seems to provide performance comparable to the iLQG method presented in this work. It is possible that the cost shaping methods which were employed in this work (e.g. the maximum likelihood assumption used for the time-to-impact) greatly impeded the performance of the iLQG algorithm. Therefore, it is desirable in future work to develop methodology which more accurately approximates the solution to the true system.

Part of the failure of the iLQG controller to achieve close to real-time performance may also be attributable to the cost shaping methods which were employed

in this work. Since divergence would occur for large step-sizes from the nominal trajectory, rather restrictive control of the step-size was implemented. This resulted in long convergence times which may be avoidable with improved cost shaping.

Similar to the outfielder problem, trajectory optimization methods with quadratic costs have also been applied to missile defense [71], which makes the application of belief iLQG natural to account for the stochastic component of the problem. Unlike the fielders studied in this work, a missile interceptor would not be constrained to motion in the ground plane. Thus, the interceptor would need to determine a trajectory that provides the highest probability of interception in 3-dimensional space rather than a plane, as is considered in this work. Since the positions of the target and the interceptor through 3-dimensional space vary rapidly with time, incorporating time-to-impact uncertainty into planning is crucial to maximizing the performance of the interceptor [100]. A ballistic missile defense system also has access to external sensing (e.g. ground-based radar [122]), which may be incorporated into the planner. Multiple interceptors may also coordinate to create a network of active sensors, allowing a belief space planner to coordinate the actions of each interceptor to maximize the probability of interception. Due to extreme time-sensitivity and high velocities that are involved, time delays must also be accounted for, which were not modeled in this work. Active ballistic missiles are also capable of performing avoidance maneuvers, so game theoretical modeling would be necessary for optimal behavior [95]. The application of the methods presented in this work to such game theoretic problems may be an area of future research.

Heuristics have also been applied to missile target interception with the intent of achieving fast and reliable performance, similar to the catching heuristics studied in this

work. For example, Proportional Navigation (PN, [71]) operates based on the intuition that if interceptor's line-of-sight remains fixed on the target then interception will occur if the interceptor is faster and more maneuverable than the target, and thus PN has been a widely studied interception heuristic. However, as the capabilities of the target increase, more sophisticated algorithms are needed for successful interception, making optimal control approaches more suitable [71]. The persistent need of missile defense systems to make timely decisions would benefit from methodology to generate new heuristics which make fast and reliable decisions in these particularly demanding systems, which is especially true for the interception of ballistic missiles during the boost and midcourse phases.

Heuristic controllers, which are specifically designed to exploit the system model in order to provide reliable results efficiently, are advantageous in that they may quickly arrive at a decision, and they may also operate using limited information [56][61], although this point was not studied in this work. However, the greatest benefit from the study of heuristics may be derived from the human thought process that develops the heuristic. Humans are experts at developing reliable behavior while simultaneously seeking to limit the resource demands of an agent. Through more extensive study of these mechanisms of human thought, it may be possible to enable artificial agents to similarly develop fast and reliable heuristics.

REFERENCES

[1] Aberdeen, D., & Baxter, J. (2002). Scaling internal-state policy-gradient methods for POMDPs. In International Conference on Machine Learning, Sydney, Australia.

[2] Agha-mohammadi, A., Agarwal, S., Kim, S.-K., Chakravorty, S., and Amato, N. M. (2018). SLAP: Simultaneous localization and planning under uncertainty via dynamic replanning in belief space. *IEEE Transactions on Robotics*, 34(5):1195–1214

[3] Agha-mohammadi, A., Agarwal, S., Mahadevan, A., Chakravorty, S., Tomkins, D., Denny, J., and Amato, N. (2014). Robust online belief space planning in changing environments: Application to physical mobile robots. In *IEEE International Conference on Robotics and Automation*, pages 149–156.

[4] Agha-Mohammadi, A., Chakravorty, S. and Amato, N. (2014) FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *The International Journal of Robotics Research* 33(2): 268–304.

[5] Alterovitz, R., Simeon, T., Goldberg, K. (2007). The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In *Robotics: Science and Systems*, pages 1–9.

[6] Anderson, M. L. (2003). Embodied cognition: A field guide. *Artificial intelligence*, *149*(1), 91-130.

[7] Athans, M. (1971). The role and use of the stochastic linear-quadratic-Gaussian problem in control system design. *IEEE transactions on automatic control*, *16*(6), 529-552.

[8] Babler, T. G., & Dannemiller, J. L. (1993). Role of image acceleration in judging landing location of free-falling projectiles. *Journal of Experimental Psychology: Human Perception and Performance*, *19*(1), 15.

[9] Bajracharya, M., Maimone, M. W., & Helmick, D. (2008). Autonomy for mars rovers: Past, present, and future. *Computer*, *41*(12).

[10] Bäuml, B., Wimböck, T., & Hirzinger, G. (2010, October). Kinematically optimal catching a flying ball with a hand-arm-system. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on* (pp. 2592-2599). IEEE.

[11] Belousov, B., Neumann, G., Rothkopf, C. A., & Peters, J. R. (2016). Catching heuristics are optimal control policies. In *Advances in Neural Information Processing Systems* (pp. 1426-1434).

[12] Birbach, O., Frese, U., & Bäuml, B. (2011, May). Realtime perception for catching a flying ball with a mobile humanoid. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on* (pp. 5955-5962). IEEE.

[13] Boddy, M., & Dean, T. L. (1994). Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, *67*(2), 245-285.

[14] Bonet, B. (2002). An epsilon-optimal grid-based algorithm for partially observable Markov decision processes. In International Conference on Machine Learning, pp. 51–58, Sydney, Australia. Morgan Kaufmann.

[15] Brafman, R. I. (1997). A heuristic variable grid solution method for POMDPs. In Proc. of the National Conference on Artificial Intelligence.

[16] Brancazio, P. J. (1985). Looking into Chapman's homer: The physics of judging a fly ball. *American Journal of Physics*, *53*, 849-855.

[17] Brechtel, S., Gindele, T., & Dillmann, R. (2013, February). Solving continuous POMDPs: Value iteration with incremental learning of an efficient space representation. In *International Conference on Machine Learning* (pp. 370-378).

[18] Brooks, A. (2007). Parametric POMDPs for Planning in Continuous State Spaces. *Ph.D. Thesis*, The University of Sydney.

[19] Bry, A., & Roy, N. (2011, May). Rapidly-exploring random belief trees for motion planning under uncertainty. In *2011 IEEE international conference on robotics and automation* (pp. 723-730). IEEE.

[20] Campbell, M., Egerstedt, M., How, J. P., & Murray, R. M. (2010). Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, *368*(1928), 4649-4672.

[21] Chapman, S. (1968). Catching a baseball. *American Journal of Physics*, *36*, 868-870.

[22] Cheney, W., & Kincaid, D. (2009). Linear algebra: Theory and applications. *The Australian Mathematical Society*, *110*.

[23] Corke, P. I. (1996). *Visual Control of Robots: high-performance visual servoing* (pp. 136-7). Taunton, UK: Research Studies Press.

[24] Crisan, D., & Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on signal processing*, *50*(3), 736-746.

[25] Dallaire, P. Besse, C., Ross, S. & Chaib-draa, B. (2009). Bayesian reinforcement learning in continuous POMDPs with Gaussian processes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[26] Dannemiller, J. L., Babler, T. G., & Babler, B. L. (1996). On Catching Fly Balls. *Science, 273,* 256-257.

[27] Deguchi, K., Sakurai, H., & Ushida, S. (2008, September). A goal oriented just-in-time visual servoing for ball catching robot arm. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (pp. 3034-3039). IEEE.

[28] Deisenroth, M., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)* (pp. 465-472).

[29] Dienes, Z., & McLeod, P. (1993). How to catch a cricket ball. *Perception*, *22*(12), 1427-1439.

[30] Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous localization and mapping: part I. *IEEE robotics & automation magazine*, *13*(2), 99-110.

[31] Erez, T., & Smart, W. (2010). A scalable method for solving high-dimensional continuous pomdps using local approximation. in *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*.

[32] Fink, P. W., Foo, P. S., & Warren, W. H. (2009). Catching fly balls in virtual reality: A critical test of the outfielder problem. *Journal of vision*, *9*(13), 14-14.

[33] Frese, U., Bauml, B., Haidacher, S., Schreiber, G., Schäfer, I., Hahnle, M., & Hirzinger, G. (2001). Off-the-shelf vision for a robotic ball catcher. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*(Vol. 3, pp. 1623-1629). IEEE.

[34] Gayanov, R., Mironov, K., Mukhametshin, R., Vokhmintsev, A., & Kurennov, D. (2018). Transportation of small objects by robotic throwing and catching: applying genetic programming for trajectory estimation. *IFAC-PapersOnLine*, *51*(30), 533-537.

[35] Gigerenzer, G., & Selten, R. (Eds.). (2002). *Bounded rationality: The adaptive toolbox*. MIT press.

[36] Guez, A., & Pineau, J. (2010, May). Multi-tasking SLAM. In *2010 IEEE International Conference on Robotics and Automation* (pp. 377-384). IEEE.

[37] Hansen, E. A. (1998, July). Solving POMDPs by searching in policy space. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 211-219). Morgan Kaufmann Publishers Inc.

[38] Höfer, S. (2017). On decomposability in robot reinforcement learning. Ph.D. Thesis. Technischen Universität Berlin.

[39] Horvitz, E. (1988, August). Reasoning under Varying and Uncertain Resource Constraints. In *AAAI* (Vol. 88, pp. 111-116).

[40] Hsu, D., Lee, W. S., & Rong, N. (2008). What makes some POMDP problems easy to approximate?. In *Advances in neural information processing systems* (pp. 689-696).

[41] Jacobs, T. M., Lawrence, M.D., Hong, K., Giordano, N., & Giordano, N. (1996). On Catching Fly Balls. *Science*, *273*, 257-258.

[42] Julier, S. J., & Uhlmann, J. K. (1997, July). New extension of the Kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI* (Vol. 3068, pp. 182-194). International Society for Optics and Photonics.

[43] Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence*, *101*(1-2), 99-134.

[44] Kaelbling, L. P., & Lozano-Pérez, T. (2013). Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, *32*(9-10), 1194-1227.

[45] Kahneman, D. (2011). *Thinking, fast and slow*. Macmillan.

[46] Karaman, S., & Frazzoli. E. (2010) Incremental sampling-based algorithms for optimal motion planning. Robotics: Science and Systems.

[47] Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, *12*(4), 566-580.

[48] Kirsh, D., & Maglio, P. (1994). On distinguishing epistemic from pragmatic action. *Cognitive science*, *18*(4), 513-549.

[49] Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, *22*(1), 79-86.

[50] Kurniawati, H., Hsu, D., & Lee, W. S. (2008, June). Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Robotics: Science and systems* (Vol. 2008).

[51] Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and brain sciences*, *40*.

[52] Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. (2016). Continuous control with deep

reinforcement learning. In *Proceedings of International Conference of Learning Representations.*

[53] Lippiello, V., Ruggiero, F., & Siciliano, B. (2013). 3D monocular robotic ball catching. *Robotics and Autonomous Systems*, *61*(12), 1615-1625.

[54] Linderoth, M. (2013). *On Robotic Work-Space Sensing and Control*. Ph.D. Thesis. Lund Institute of Technology, Lund University.

[55] Marken, R. S. (2001). Controlled variables: Psychology as the center fielder views it. *The American Journal of Psychology*, *114*(2), 259.

[56] McBeath, M. K., Shaffer, D. M., & Kaiser, M. K. (1995). How baseball outfielders determine where to run to catch fly balls. *Science*, *268*(5210), 569-573.

[57] McBeath, M. K., Shaffer, D. M., & Kaiser, M. K. (1996). On Catching Fly Balls. *Science, 273,* 258-260.

[58] McBeath, M. K., Nathan, A. M., Bahill, A. T., & Baldwin, D. G. (2008). Paradoxical popups: Why are they hard to catch?. *arXiv preprint arXiv:0803.4357*.

[59] McLeod, P., Reed, N., & Dienes, Z. (2001). Toward a unified fielder theory: What we do not yet know about how people run to catch a ball. *Journal of Experimental Psychology: Human Perception and Performance*, *27*(6), 1347.

[60] McLeod, P., Reed, N., & Dienes, Z. (2002). The optic trajectory is not a lot of use if you want to catch the ball. *Journal of Experimental Psychology: Human Perception and Performance*, *28*(6), 1499.

[61] McLeod, P., Reed, N., & Dienes, Z. (2006). The generalized optic acceleration cancellation theory of catching. *Journal of Experimental Psychology: Human Perception and Performance*, *32*(1), 139.

[62] Michaels, C. F., & Oudejans, R. R. (1992). The optics and actions of catching fly balls: Zeroing out optical acceleration. *Ecological Psychology*, *4*(4), 199-222.

[63] Miller, S., Harris, A., & Chong, E. (2009). Coordinated guidance of autonomous uavs via nominal belief-state optimization. in *American Control Conference*, pp. 2811–2818.

[64] Miyazaki, F., & Mori, R. (2004, March). Realization of ball catching task using a mobile robot. In *Networking, Sensing and Control, 2004 IEEE International Conference on* (Vol. 1, pp. 58-63). IEEE.

[65] Moré, J. J. (1978). The Levenberg-Marquardt algorithm: Implementation and Theory. In *Numerical analysis* (pp. 105-116). Springer, Berlin, Heidelberg.

[66] Mu, B., Agha-mohammadi, A., Paull, L., Graham, M., How, J., and Leonard, J. (2017). Two-stage focused inference for resource-constrained collision-free navigation. IEEE Transactions on Robotics, 33(1):124– 140.

[67] Mu, B., Giamou, M., Paull, L., Agha-mohammadi, A., Leonard, J., and How, J. (2016). Information-based active slam via topological feature graphs. In IEEE Conference on Decision and Control, pages 5583–5590.

[68] Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML* (Vol. 99, pp. 278-287).

[69] Ng, A. Y., & Jordan, M. (2000, June). PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence* (pp. 406-415). Morgan Kaufmann Publishers Inc.

[70] Otsu, K., Agha-mohammadi, A., and Paton, M. (2018). Where to Look? Predictive perception with applications to planetary exploration. IEEE Robotics and Automation Letters, 3(2):635–642.

[71] Palumbo, N. F., Blauwkamp, R. A., & Lloyd, J. M. (2010). Modern homing missile guidance theory and techniques. *Johns Hopkins APL technical digest*, *29*(1), 42-59.

[72] Paquet, S., Chaib-draa, B., & Ross, S. (2006). Hybrid POMDP algorithms. In Proceedings of *The Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains* (MSDM-06), pp. 133–147.

[73] Patil, S., Kahn, G., Laskey, M., Schulman, J., Goldberg, K., & Abbeel, P. (2015). Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation. In *Algorithmic foundations of robotics XI* (pp. 515-533). Springer, Cham.

[74] Pearl, J. (1984). *Heuristics – Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.

[75] Petersen, K. B., & Pedersen, M. S. (2008). The matrix cookbook. *Technical University of Denmark*, *7*(15), 510.

[76] Pineau, J., Roy, N., & Thrun, S. (2001, June). A hierarchical approach to POMDP planning and execution. In *Workshop on hierarchy and memory in reinforcement learning (ICML)* (Vol. 65, No. 66, p. 51).

[77] Pineau, J., Gordon, G., & Thrun, S. (2006). Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, *27*, 335-380.

[78] Platt, R., Tedrake, R., Kaelbling, L., and Lozano-Perez, T. (2010) Belief Space Planning assuming Maximum Likelihood Observations. In Robotics: Science and Systems (RSS).

[79] Platt, R.; Kaelbling, L.; Lozano-Perez, T.; and Tedrake, R. (2011) Efficient Planning in Non-Gaussian Belief Spaces and its Application to Robot Grasping. In Int. Symp. on Robotics Research (ISRR).

[80] Porta, J. M., Vlassis, N., Spaan, M. T., & Poupart, P. (2006). Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research*, *7.* (Nov), 2329-2367.

[81] Powers, W. T. (1973). *Behavior: The control of perception.* Chicago: Aldine.

[82] Poupart, P., & Boutilier, C. (2003). Value-directed compression of POMDPs. In *Advances in neural information processing systems* (pp. 1579-1586).

[83] Poupart, P., & Boutilier, C. (2004). Bounded finite state controllers. In *Advances in neural information processing systems* (pp. 823-830).

[84] Prentice, S., & Roy, N. (2009). The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, *28*(11-12), 1448-1465.

[85] Rhudy, M. B. (2013). Sensitivity and stability analysis of nonlinear Kalman filters with application to aircraft attitude estimation. *Ph.D. Thesis*, West Virginia University.

[86] Ross, S., Pineau, J., Paquet, S., & Chaib-Draa, B. (2008). Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, *32*, 663-704.

[87] Roy, N., Gordon, G., & Thrun, S. (2005). Finding approximate POMDP solutions through belief compression. *Journal of artificial intelligence research*, *23*, 1-40.

[88] Saxberg, B. V. (1987a). Projected free fall trajectories I. Theory and Simulation. *Biological Cybernetics*, *56*(2-3), 159-175.

[89] Saxberg, B. V. (1987b). Projected free fall trajectories II. Human Experiments. *Biological Cybernetics*, *56*(2-3), 176-187.

[90] Shaffer, D. M., & McBeath, M. K. (2002). Baseball outfielders maintain a linear optical trajectory when tracking uncatchable fly balls. *Journal of Experimental Psychology: Human Perception and Performance*, *28*(2), 335.

[91] Shaffer, D. M., McBeath, M. K., Roy, W. L., & Krauchunas, S. M. (2003). A linear optical trajectory informs the fielder where to run to the side to catch fly balls. *Journal of Experimental Psychology: Human Perception and Performance*, *29*(6), 1244.

[92] Shaffer, D. M., Krauchunas, S. M., Eddy, M., & McBeath, M. K. (2004). How dogs navigate to catch Frisbees. *Psychological Science*, *15*(7), 437-441.

[93] Shaffer, D. M., & McBeath, M. K. (2005). Naive beliefs in baseball: systematic distortion in perceived time of apex for fly balls. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *31*(6), 1492.

[94] Shibata, K., Nishino, T., & Okabe, Y. (1995). Active perception based on reinforcement learning. In *Proc. of WCNN* (Vol. 95, pp. 170-173).

[95] Shinar, J., & Shima, T. (1996, December). A game theoretical interceptor guidance law for ballistic missile defence. In *Proceedings of 35th IEEE Conference on Decision and Control* (Vol. 3, pp. 2780-2785). IEEE.

[96] Silver, D., & Veness, J. (2010). Monte-Carlo planning in large POMDPs. In *Advances in neural information processing systems* (pp. 2164-2172).

[97] Simon, D. (2006) *Optimal State Estimation*, Wiley, New York.

[98] Simon, H. A. (1955). A behavioral model of rational choice. *The quarterly journal of economics*, *69*(1), 99-118.

[99] Smith, T., & Simmons, R. (2004, July). Heuristic search value iteration for POMDPs. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence* (pp. 520-527). AUAI Press.

[100] Snyder, M. (2016). A New Path Planning Guidance Law For Improved Impact Time Control of Missiles and Precision Munitions. *Ph.D. thesis*, University of Central Florida.

[101] Sondik, E. (1971). The Optimal Control of Partially Observable Markov Processes. *Ph.D. thesis*, Stanford University.

[102] Spaan, M. T., & Vlassis, N. (2005). Perseus: Randomized point-based value iteration for POMDPs. *Journal of artificial intelligence research*, *24*, 195-220.

[103] Srinivasan, M. V., Zhang, S. W., Chahl, J. S., Barth, E., & Venkatesh, S. (2000). How honeybees make grazing landings on flat surfaces. Biological cybernetics, 83(3), 171-183.

[104] Stachniss, C., Grisetti, G., & Burgard, W. (2005, June). Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems* (Vol. 2, pp. 65-72).

[105] Strader, J., Otsu, K., & Agha-mohammadi, A. A. (2019). Perception-aware autonomous mast motion planning for planetary exploration rovers. *Journal of Field Robotics*.

[106] Stewart, J. (2010). *Essential calculus*. Cengage Learning.

[107] Sugar, T. G., McBeath, M. K., Suluh, A., & Mundhra, K. (2006). Mobile robot interception using human navigational principles: Comparison of active versus passive tracking algorithms. *Autonomous Robots*, *21*(1), 43-54.

[108] Sun, W., van den Berg, J., & Alterovitz, R. (2016). Stochastic extended LQR for optimization-based motion planning under uncertainty. *IEEE Transactions on Automation Science and Engineering*, *13*(2), 437-447.

[109] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Unpublished Manuscript.

[110] Thrun, S. (2000). Monte Carlo POMDPs. In *Advances in neural information processing systems* (pp. 1064-1070).

[111] Todd, J. T. (1981). Visual information about moving objects. *Journal of Experimental Psychology: Human Perception and Performance*, *7*(4), 795.

[112] Todorov, E., & Li, W. (2005, June). A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference, 2005.* (pp. 300-306). IEEE.

[113] Toussaint, M., Charlin, L., & Poupart, P. (2008, July). Hierarchical POMDP Controller Optimization by Likelihood Maximization. In *UAI* (Vol. 24, pp. 562-570).

[114] Tresilian, J. R. (1995). Study of a servo-control strategy for projectile interception. *The quarterly journal of experimental psychology*, *48*(3), 688-715.

[115] Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *science*, *185*(4157), 1124-1131.

[116] Valencia, R., Morta, M., Andrade-Cetto, J., & Porta, J. M. (2013). Planning reliable paths with pose SLAM. *IEEE Transactions on Robotics*, *29*(4), 1050-1059.

[117] van den Berg, J., Patil, S., and Alterovitz, R. (2011). Motion Planning under Uncertainty using Differential Dynamic Programming in Belief Space. In Int. Symp. on Robotics Research (ISRR).

[118] van den Berg, J., Patil, S., & Alterovitz, R. (2012, July). Efficient Approximate Value Iteration for Continuous Gaussian POMDPs. In *AAAI*.

[119] van den Berg, J., Patil, S., & Alterovitz, R. (2012). Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, *31*(11), 1263-1278.

[120] Wang, W., McBeath, M. K., & Sugar, T. G. (2015). Navigational strategy used to intercept fly balls under real-world conditions with moving visual background fields. *Attention, Perception, & Psychophysics*, *77*(2), 613-625.

[121] Welch, G., and Bishop, G. 2006. An Introduction to the Kalman Filter. Technical Report TR 95-041, Univ. North Carolina at Chapel Hill.

[122] Wilkening, D. A. (2004). Airborne boost-phase ballistic missile defense. *Science and Global Security*, *12*(1-2), 1-67.

[123] Wilson, M. (2002). Six views of embodied cognition. *Psychonomic bulletin & review*, *9*(4), 625-636.

[124] Wilson, A. D., & Golonka, S. (2013). Embodied cognition is not what you think it is. *Frontiers in psychology*, *4*, 58.

[125] Yeadon, M. R., King, M. A., & Wilson, C. (2006). Modeling the maximum voluntary joint torque/angular velocity relationship in human movement. *Journal of biomechanics*, *39*(3), 476-482.

[126] Zarchan, P. (1999). Ballistic missile defense guidance and control issues. *Science & Global Security*, *8*(1), 99-124.

[127] Zhou, R., & Hansen, E. A. (2001, August). An improved grid-based approximation algorithm for POMDPs. In *IJCAI* (pp. 707-716).

[128] Zhou, E., Fu, M. C., & Marcus, S. I. (2010). Solving continuous-state POMDPs via density projection. *IEEE Transactions on Automatic Control*, *55*(5), 1101-1116.

APPENDIX A: BELIEF iLQG

The modified belief space variant of iLQG presented in Chapter 4 is based on the work of van den Berg et al. [118], which is summarized here. The problem statement is the same as presented in Section 4.1, so that the value function $v_k[\hat{\boldsymbol{x}}_k, P_k]$ at time k is approximated by a function that is quadratic in the mean and linear in the variance that is locally valid around some nominal belief $\bar{\boldsymbol{x}}_k, \bar{P}_k$; although the variance term in the value function is expressed as a vectorized matrix:

$$v_k[\hat{\boldsymbol{x}}, P] \approx s_k + \frac{1}{2}(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k)^T S_k(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \boldsymbol{s}_k^T(\hat{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_k) + \boldsymbol{t}_k^T \mathrm{vec}[P_k - \bar{P}_k] \tag{169}$$

where at the final time-step $k = \ell$,

$$\begin{aligned} S_\ell &= \frac{\partial^2 c_\ell}{\partial \hat{\boldsymbol{x}}_\ell \partial \hat{\boldsymbol{x}}_\ell}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], & \boldsymbol{s}_\ell^T &= \frac{\partial c_\ell}{\partial \hat{\boldsymbol{x}}_\ell}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], \\ s_\ell &= c_\ell[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], & \boldsymbol{t}_\ell^T &= \frac{\partial c_\ell}{\partial \mathrm{vec}[P_\ell]}[\bar{\boldsymbol{x}}_\ell, \bar{P}_\ell], \end{aligned} \tag{170}$$

The approximation of the value function at time-steps $0 \leq k < \ell$ are found by approximating the Bellman back-propagation of the approximate value function at time-step $k + 1$:

$$
\begin{aligned}
v_k[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] &= \min_{\boldsymbol{u}}\left(c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + \mathrm{E}\left[v_{k+1}[\hat{\boldsymbol{x}}_{k+1}, \boldsymbol{u}_{k+1}]\right]\right) \\
&\approx \min_{\boldsymbol{u}}\Bigg(c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \\
&\quad + \mathrm{E}\Bigg[s_{k+1} + \frac{1}{2}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w} - \overline{\boldsymbol{x}}_{k+1})^T S_{k+1}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w} - \overline{\boldsymbol{x}}_{k+1}) \\
&\quad + \boldsymbol{s}_{k+1}^T(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] + \boldsymbol{w} - \overline{\boldsymbol{x}}_{k+1}) + \boldsymbol{t}_{k+1}^T \mathrm{vec}[\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1}]\Bigg]\Bigg) \\
&\approx \min_{\boldsymbol{u}}\Bigg(c_k[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] + s_{k+1} \\
&\qquad + \frac{1}{2}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1})^T S_{k+1}(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1}) \\
&\qquad + \boldsymbol{s}_{k+1}^T(\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1}) \\
&\qquad + \boldsymbol{t}_{k+1}^T \mathrm{vec}[\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1}] \\
&\qquad + \frac{1}{2}\mathrm{vec}[S_{k+1}]^T \mathrm{vec}\left[W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]\right]\Bigg)
\end{aligned}
\tag{171}
$$

Where the following identities were employed to derive the last term of Equation 171:

$$
\mathrm{E}[\boldsymbol{z}^T A \boldsymbol{z}] = \mathrm{E}[\boldsymbol{z}^T] A \mathrm{E}[\boldsymbol{z}] + \mathrm{tr}\left[A\ \mathrm{Var}[\boldsymbol{z}]\right] \tag{172}
$$

$$
\mathrm{tr}[AZ] = \mathrm{vec}[A^T]^T \mathrm{vec}[Z] \tag{173}
$$

The following first order approximations were used to further simplify Equation 171:

$$
\boldsymbol{f}[\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k] - \overline{\boldsymbol{x}}_{k+1} \approx F_k(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + G_k(\boldsymbol{u}_k - \overline{\boldsymbol{u}}_k) \tag{174}
$$

$$
\mathrm{vec}[\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] - \bar{P}_{k+1}] \approx T_k(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + U_k \mathrm{vec}[P_k - \bar{P}_k] + V_k(\boldsymbol{u}_k - \overline{\boldsymbol{u}}_k) \tag{175}
$$

$$
\mathrm{vec}\left[W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]\right] \approx \boldsymbol{y}_k + X_k(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + Y_k \mathrm{vec}[P_k - \bar{P}_k] + Z_k(\boldsymbol{u}_k - \overline{\boldsymbol{u}}_k) \tag{176}
$$

where

$$
\begin{aligned}
F_k &= \frac{\partial \boldsymbol{f}}{\partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \overline{\boldsymbol{u}}_k], & G_k &= \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \overline{\boldsymbol{u}}_k], \\
T_k &= \frac{\partial \mathrm{vec}[\Phi]}{\partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & U_k &= \frac{\partial \mathrm{vec}[\Phi]}{\partial \mathrm{vec}[P_k]}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
V_k &= \frac{\partial \mathrm{vec}[\Phi]}{\partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & X_k &= \frac{\partial \mathrm{vec}[W]}{\partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
Y_k &= \frac{\partial \mathrm{vec}[W]}{\partial \mathrm{vec}[P_k]}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & Z_k &= \frac{\partial \mathrm{vec}[W]}{\partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
\boldsymbol{y}_k &= \mathrm{vec}\big[W[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k]\big],
\end{aligned}
\tag{177}
$$

The immediate cost function $c_k[\widehat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ was approximated by a second-order approximation with respect to the mean and a first-order approximation with respect to the variance:

$$
\begin{aligned}
c_k[\widehat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] \approx q_k + \frac{1}{2}\begin{bmatrix}\widehat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix}^T \begin{bmatrix}Q_k & J_k^T \\ J_k & R_k\end{bmatrix}\begin{bmatrix}\widehat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix} + \begin{bmatrix}\boldsymbol{q}_k \\ \boldsymbol{r}_k\end{bmatrix}^T \begin{bmatrix}\widehat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k\end{bmatrix} \\
+ \boldsymbol{p}_k^T \mathrm{vec}[P_k - \bar{P}_k]
\end{aligned}
\tag{178}
$$

where

$$
\begin{aligned}
Q_k &= \frac{\partial^2 c_k}{\partial \widehat{\boldsymbol{x}}_k \partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & \boldsymbol{q}_k^T &= \frac{\partial c_k}{\partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
R_k &= \frac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & \boldsymbol{r}_k^T &= \frac{\partial c_k}{\partial \boldsymbol{u}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
J_k &= \frac{\partial^2 c_k}{\partial \boldsymbol{u}_k \partial \widehat{\boldsymbol{x}}_k}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], & q_k &= c_k[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k], \\
\boldsymbol{p}_k^T &= \frac{\partial c_k}{\partial \mathrm{vec}[P_k]}[\overline{\boldsymbol{x}}_k, \bar{P}_k, \overline{\boldsymbol{u}}_k],
\end{aligned}
\tag{179}
$$

Substituting Equations 174-179 into Equation 171, it can be seen that the value function at time k may be approximated as

$$v_k[\hat{\boldsymbol{x}}_k, P_k] \approx \min_{\boldsymbol{u}} \left(e_k + \frac{1}{2} \begin{bmatrix} \hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k \end{bmatrix}^T \begin{bmatrix} C_k & E_k^T \\ E_k & D_k \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{c}_k \\ \boldsymbol{d}_k \end{bmatrix}^T \begin{bmatrix} \hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k \\ \boldsymbol{u}_k - \overline{\boldsymbol{u}}_k \end{bmatrix} + \boldsymbol{e}_k^T \mathrm{vec}[P_k - \bar{P}_k] \right) \tag{180}$$

where

$$\begin{aligned} C_k &= Q_k + F_k^T S_{k+1} F_k, & \boldsymbol{c}_k^T &= \boldsymbol{q}_k^T + \boldsymbol{s}_{k+1}^T F_k + \boldsymbol{t}_{k+1}^T T_k + \frac{1}{2} \mathrm{vec}[S_{k+1}]^T X_k, \\ D_k &= R_k + G_k^T S_{k+1} G_k, & \boldsymbol{d}_k^T &= \boldsymbol{r}_k^T + \boldsymbol{s}_{k+1}^T G_k + \boldsymbol{t}_{k+1}^T V_k + \frac{1}{2} \mathrm{vec}[S_{k+1}]^T Z_k, \\ E_k &= J_k + G_k^T S_{k+1} F_k, & \boldsymbol{e}_k^T &= \boldsymbol{p}_k^T + \boldsymbol{t}_{k+1}^T U_k + \frac{1}{2} \mathrm{vec}[S_{k+1}]^T Y_k, \\ & & e_k &= q_k + s_{k+1} + \frac{1}{2} \mathrm{vec}[S_{k+1}]^T \boldsymbol{y}_k. \end{aligned} \tag{181}$$

A locally optimal policy $\boldsymbol{\pi}_k[\hat{\boldsymbol{x}}_k, P_k]$ at time step k can be found by referencing Equation 180 and setting the first derivative of $v_k[\hat{\boldsymbol{x}}_k, P_k]$ with respect to $\boldsymbol{u}_k$ equal to zero and then solving for $\boldsymbol{u}_k$.

$$\boldsymbol{u}_k = L_k(\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + \boldsymbol{l}_k + \overline{\boldsymbol{u}}_k \tag{182}$$

where

$$L_k = -D_k^{-1} E_k \tag{183}$$

$$\boldsymbol{l}_k = -D_k^{-1} \boldsymbol{d}_k \tag{184}$$

By substitution of Equations 182-184 into Equation 180, the desired form of the value function approximation is obtained at time step k.

$$v_k[\hat{\boldsymbol{x}}, P] \approx s_k + \frac{1}{2} (\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k)^T S_k (\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + \boldsymbol{s}_k^T (\hat{\boldsymbol{x}}_k - \overline{\boldsymbol{x}}_k) + \boldsymbol{t}_k^T \mathrm{vec}[P_k - \bar{P}_k] \tag{185}$$

where

$$S_k = C_k + L_k^T E_k, \qquad \boldsymbol{s}_k^T = \boldsymbol{c}_k^T + \boldsymbol{l}_k^T E_k,$$
$$s_k = e_k + \frac{1}{2}\boldsymbol{d}_k^T \boldsymbol{l}_k, \qquad \boldsymbol{t}_k^T = \boldsymbol{e}_k^T, \tag{186}$$

The process for iterating to a locally optimal policy is the same in both approaches.

It can be seen that the only changes made to [118] in this work (see Chapter 4) relate to the handling of the variance term. However, the method in [118] requires the partial derivative with respect to each element of P_k to calculate the matrices U_k and Y_k. Each partial derivative requires $O[n^2]$ time to evaluate, and there are n^2 elements in P_k, so in total $O[n^4]$ time is required to differentiate with respect to every element, which results in U_k and Y_k being $n^2 \times n^2$ matrix. Then, this pair of $n^2 \times n^2$ matrices are each multiplied with a vector of dimension n^2, which also requires $O[n^4]$ time for each multiplication. These products from van den Berg et al.'s [118] method are calculated in $O[n^3]$ time in this work.

APPENDIX B: NOTES ON ANALYTIC DERIVATIVES

It was suggested in Section 4.4.1 that the vectors $\boldsymbol{a}_k$, $\boldsymbol{b}_k$, $\boldsymbol{t}_k$, and $\boldsymbol{v}_k$ may be calculated numerically or analytically. Detailed here are the necessary derivatives for analytical calculation of these vectors. The variable z_i may be replaced by $\hat{x}_i$ or u_i to find the appropriate derivatives. In order to determine the analytic derivatives, it is necessary to compute the partial derivatives of F_k, H_k, M_k, and N_k with respect to each element $\hat{x}_i$ and u_i. First, the partial derivative of $\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ may be determined by

$$\left.\frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} = -\bar{P}_{k+1} \left.\frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]^{-1}}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{P}_{k+1} \tag{187}$$

where $\bar{P}_{k+1}$ is given by Equations 34-35. The derivative of $\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]^{-1}$ may be found with reference to Equation 20.

$$\begin{aligned} \left.\frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]^{-1}}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} &= \left.\frac{\partial}{\partial z_i}\left({\Gamma_k}^{-1} + H_k^T N_k^{-1} H_k\right)\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \\ &= \left.\frac{\partial {\Gamma_k}^{-1}}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} + \left.\frac{\partial}{\partial z_i} H_k^T N_k^{-1} H_k\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \end{aligned} \tag{188}$$

The first partial derivative in Equation 188 is given by

$$\left.\frac{\partial {\Gamma_k}^{-1}}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} = -\bar{\Gamma}_k^{-1} \left.\frac{\partial \Gamma_k}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{\Gamma}_k^{-1} \tag{189}$$

where

$$\begin{aligned} \left.\frac{\partial \Gamma_k}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} &= \left.\frac{\partial}{\partial z_i}\left(F_k P_k F_k^T + M_k\right)\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \\ &= \left.\frac{\partial F_k}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{P}_k \bar{F}_k^T + \bar{F}_k \bar{P}_k \left.\frac{\partial F_k^T}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} + \left.\frac{\partial M_k}{\partial z_i}\right|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \end{aligned} \tag{190}$$

The second partial derivative in Equation 188 may be expressed as

$$\frac{\partial}{\partial z_i} H_k^T N_k^{-1} H_k \Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

$$= \frac{\partial H_k^T}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{N}_k^{-1} \bar{H}_k + \bar{H}_k^T \frac{\partial N_k^{-1}}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{H}_k \tag{191}$$

$$+ \bar{H}_k^T \bar{N}_k^{-1} \frac{\partial H_k}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

The necessary partial derivatives of $W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ may be found using

$$\frac{\partial W[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} = \frac{\partial}{\partial z_i} K_k H_k \Gamma_k \Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

$$= \frac{\partial K_k}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} H_k \Gamma_k + K_k \frac{\partial H_k}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \Gamma_k \tag{192}$$

$$+ K_k H_k \frac{\partial \Gamma_k}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

where the partial derivative of Γ_k was determined in Equation 190, and the partial derivative of H_k is assumed to be known. The partial derivative with respect to K_k may be found with the aid of Equation 22.

$$\frac{\partial K_k}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} = \frac{\partial}{\partial z_i} \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k] H_k^T N_k^{-1} \Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

$$= \frac{\partial \Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{H}_k^T \bar{N}_k^{-1} + \bar{P}_{k+1} \frac{\partial H_k^T}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k} \bar{N}_k^{-1} \tag{193}$$

$$+ \Phi[\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k] \bar{H}_k^T \frac{\partial N_k^{-1}}{\partial z_i}\Big|_{\bar{\boldsymbol{x}}_k, \bar{P}_k, \bar{\boldsymbol{u}}_k}$$

where the partial derivative of $\Phi[\hat{\boldsymbol{x}}_k, P_k, \boldsymbol{u}_k]$ was determined in Equations 187-192 , and the partial derivative of H_k is assumed to be known. If necessary, the partial derivative of

N_k^{-1} may be found with the aid of the identity in Equation 44, given that the partial derivative of N_k is assumed to be known.

APPENDIX C: SEPARABILITY OF FIRST-ORDER EXPANSIONS

Consider a scalar function of two scalar variables, $f[x, y]$. Take the first order Taylor expansion about $\bar{x}, \bar{y}$:

$$f[x, y] - f[\bar{x}, \bar{y}] \approx \frac{\partial f[\bar{x}, \bar{y}]}{\partial x}(x - \bar{x}) + \frac{\partial f[\bar{x}, \bar{y}]}{\partial y}(y - \bar{y}) \tag{194}$$

Now consider the expansions in which one variable is fixed.

$$f[x, \bar{y}] - f[\bar{x}, \bar{y}] \approx \frac{\partial f[\bar{x}, \bar{y}]}{\partial x}(x - \bar{x}) \tag{195}$$

$$f[\bar{x}, y] - f[\bar{x}, \bar{y}] \approx \frac{\partial f[\bar{x}, \bar{y}]}{\partial y}(y - \bar{y}) \tag{196}$$

By substitution of Equations 195-196 into Equation 194:

$$f[x, y] - f[\bar{x}, \bar{y}] \approx (f[x, \bar{y}] - f[\bar{x}, \bar{y}]) + (f[\bar{x}, y] - f[\bar{x}, \bar{y}]) \tag{197}$$

which is valid to first order. A similar argument could be constructed for vector and matrix functions with more than two variables.

APPENDIX D: IMMEDIATE COST FUNCTION CONSIDERATIONS

The actual expected immediate reward is given by the probability that the ball will be caught given a belief $\hat{\boldsymbol{x}}_t, P_t$. This may be expressed as

$$\rho[\hat{\boldsymbol{x}}_t, P_t, \boldsymbol{u}_t] = \mathrm{E}[\mathcal{R}(\boldsymbol{x}_t, \boldsymbol{u}_t)] = \int_{-\infty}^{\infty} \mathcal{R}(\boldsymbol{x}_t, \boldsymbol{u}_t) p_{\boldsymbol{x}_t}(\boldsymbol{x}_t) d\boldsymbol{x}_t \tag{198}$$

$$\boldsymbol{x}_t \sim \mathcal{N}(\hat{\boldsymbol{x}}_t, P_t)$$

where $\rho[\hat{\boldsymbol{x}}_t, P_t, \boldsymbol{u}_t]$ is the expected immediate reward of belief $\hat{\boldsymbol{x}}_t, P_t$ and input $\boldsymbol{u}_t$, and $p_{\boldsymbol{x}_t}(\boldsymbol{x}_t)$ is the probability density function for $\boldsymbol{x}_t$. From Equation 123, it can be seen that a reward of 1 is received only under the conditions in which the ball is caught. So Equation 198 may be decomposed into integration over two mutually exclusive spaces, with the value of the reward function being one in the region where the ball is caught and zero in the region in which the ball is not caught.

$$\int_{-\infty}^{\infty} \mathcal{R}(\boldsymbol{x}_t, \boldsymbol{u}_t) p_{\boldsymbol{x}_t}(\boldsymbol{x}_t) d\boldsymbol{x}_t = \int_{\substack{ball\ is \\ caught}} 1 \cdot p_{\boldsymbol{x}_t}(\boldsymbol{x}_t) d\boldsymbol{x}_t + \int_{\substack{ball\ not \\ caught}} 0 \cdot p_{\boldsymbol{x}_t}(\boldsymbol{x}_t) d\boldsymbol{x}_t \tag{199}$$

Thus, $\rho[\hat{\boldsymbol{x}}_t, P_t, \boldsymbol{u}_t]$ is the probability that the ball is caught in belief $\hat{\boldsymbol{x}}_t, P_t$, which is

$$\begin{aligned} \rho[\hat{\boldsymbol{x}}_t, P_t, \boldsymbol{u}_t] &= \Pr\left[\left(z_{b,t} = 0\right) \cap \left((\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t)^{1/2} \leq \epsilon\right)\right] \\ &= \Pr\left[(\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t)^{1/2} \leq \epsilon \,\middle|\, z_{b,t} = 0\right] \Pr\left[z_{b,t} = 0\right] \end{aligned} \tag{200}$$

by the definition of conditional probability, where the event $z_{b,t} = 0$ is interpreted as the probability that the ball lands within the time interval considered by the current time step, since the probability of landing at a particular continuous time t is zero.

The reward function given by Equation 200 is not well suited for the iLQG method, since there are beliefs in which the Hessian is indefinite and thus does not satisfy

the constraints given in Equation 16. A common approach to resolve this issue is to convert the reward function into a cost function using the negated log-probability, which yields the candidate cost function

$$\begin{aligned} c_t[\hat{\boldsymbol{x}}_t, P_t, \boldsymbol{u}_t] &= -\log\left[\Pr\left[(\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t)^{1/2} \leq \epsilon \,\middle|\, z_{b,t} = 0\right] \Pr\left[z_{b,t} = 0\right]\right] \\ &= -\log\left[\Pr\left[(\boldsymbol{x}_t^T \chi^T \chi \boldsymbol{x}_t)^{1/2} \leq \epsilon \,\middle|\, z_{b,t} = 0\right]\right] - \log\left[\Pr\left[z_{b,t} = 0\right]\right] \end{aligned} \tag{201}$$

This candidate cost function possesses a desirable positive-definite Hessian since $\boldsymbol{x}_t$ is Gaussian, however, this cost function results in undesirable behavior. The first term represents the cost associated with the distance error, given that the ball lands within the time interval considered by the current time step. This is problematic because this term incentivizes the fielder to track the most likely landing spot of the ball without regard to probability that the ball actually will land. Meanwhile, the second term assigns a large cost to states in which the probability of the ball landing within the time interval considered by the current time step is close to zero. This runs counter to intuition, which dictates that the fielder should only be penalized if the ball has a high probability of landing, but the fielder is not correctly positioned to intercept it, rather penalizing the fielder simply because the catch must be made in the future.

APPENDIX E: BALL-CATCHING ROBOTS

To date, there has not yet been any success in designing a robot that is capable of catching fly balls with anywhere near the proficiency of a human, as the maximum distances (<7 m; [12]) over which the ball has been thrown and successfully caught by a mobile robot are significantly less than those which are experienced in professional baseball, where distances of 100 m and velocities of 40 m/s are routine. Additionally, professional baseball players are exposed to ball trajectories which are shaped by much larger drag and Magnus forces, further disturbing the trajectory from the parabolic ideal.

There have been several cases in which researches have successfully implemented robotic arms on fixed mounts in the catching task. Frese et al. [33] implemented a 7 DOF arm on fixed mount with net mounted on the end effector. Stereo cameras with a 1 m baseline on a fixed external mount were used to track the ball. The ball trajectory was modeled with drag effects and was estimated using an EKF. A heuristic was used to determine the catch point comfortably within the reachable space of the robot so that it can adjust to prediction errors. The success rate was about 66% for balls tossed across a room. Deguchi et al. [27] similarly implemented a 7 DOF arm with a cup mounted on the end effector to perform catching. Stereo cameras with a large baseline were also used to track the ball, and batch estimation was used to fit a parabolic trajectory from all available images. A point was then selected along this parabolic trajectory to be the catch point. Instead of optimizing the end effector motion, visual servoing was performed in epipolar coordinates using the stereo images to move the end effector (the cup) to the desired catch point. The basis of their strategy was that visual servoing was faster to compute and more robust to modeling errors, although the success rate was not reported.

Linderoth [54] implemented an industrial robot on a fixed mount for ball catching with a success rate of about 72%. A box was mounted on the end effector with a hole cut into it that was only slightly larger than the ball, such that less than 8 mm of error was needed for the ball to be caught. Target tracking was again performed using stereo vision using a large baseline and a fixed external mount. The ball trajectory was modeled as ballistic with drag and was estimated using a multiple hypothesis tracker with an EKF for each hypothesis. The visual detection could only be done reliably within 4 m with ball velocities up to 11 m/s, so the detection, estimation, planning, and implementation had to be performed rapidly, which was ultimately accomplished with only a 44 ms delay for the compute time. The planning was performed exceptionally fast (~4 μs) by planning each joint individually in joint space. The implementation of the planned control actions could also be performed very rapidly using the ABB IRB 140 industrial robot, for which max joint velocities range from 200-450°/s, which is similar to the rate of a human knee extension [125]. Additionally, motions could also be performed very accurately, with position repeatability of 0.03 mm.

Lippiello et al. [53] implemented a monocular camera on the end effector of a fixed-mount robotic arm. The moveable monocular camera enabled the robot to employ active perception to increase the observability of the ball's trajectory, which was modeled as a ballistic trajectory with drag. The directional vectors from the camera to the ball and the camera poses were used to estimate the ball's trajectory through batch estimation with the Levenberg–Marquardt algorithm [65]. The estimated trajectory was used to determine a catch point, which was optimized to minimize torque at the arm's joints. The

configuration enabled a 90% success rate for balls tossed within reach of the robot from across a room.

In addition to fixed-mount robotic arms, there have been a few instances in which mobile robots were used for the ball catching task, which can be considered more akin to the problem considered in this research. Miyazaki and Mori [64] developed the Gaining Angle of Gaze (GAG) heuristic for use on a differential drive robot. The GAG heuristic is based on OAC for catching the ball using a differential drive robot in two dimensions with a monocular camera. The algorithm demonstrated limited ability to catch the ball over only short distances. Sugar et al. [107] developed a mobile robot employing OAC to catch balls in the sagittal plane using a monocular camera with some success, again only over short distances. The most complete ball catching robot to date has been a humanoid robot with a holonomic wheeled base that has demonstrated the ability to catch balls thrown from 5-7 m away at velocities around 7 m/s using a four-fingered hand with a success rate of about 80% [10][12]. The humanoid robot had stereo vision cameras with a short baseline mounted on its head, which induced shaking of the cameras when the robot would move. A head mounted IMU was implemented to compensate for the image noise induced by the shaking cameras. The robot performed estimation using a multiple hypothesis tracker with an Unscented Kalman Filter (UKF; [42]) for every hypothesis. The ball's trajectory was modeled with drag but no Magnus forces, which are negligible at the velocities and spin rates that were considered. The overall delay from the camera shutter to the movement being implemented was estimated to be around 90 ms, with computation time being an important factor that was considered in the design process

[10]. The main sources of error leading to failure were attributed to the visual tracking system and prediction.