ABSTRACT

COST AND RISK TRADE-OFF ANALYSIS OF OPTIMAL CONTROLLERS

by Adrianna Patch

Cost and risk are valuable performance metrics in control systems. This thesis outlines various definitions of cost and risk, and furthermore provides context in which they may be used. In this work, we develop a novel analytical solution designed to trade off cost and safety in the Linear Quadratic Gaussian (LQG) scenario. By means of comparison, we have implemented multiple existing optimal control methods in stochastic, linear applications. An existing method of particular interest to this work is the CVaR Barrier Function, which makes use of the Control Barrier Function (CBF) as well as the concept of Conditional-Value-at-Risk (CVaR) safety. In this work we conduct a cost-safety analysis on these methods in accordance to the cost and safety definitions defined in our problem formation. Finally, we demonstrate performance comparisons between our method and existing methods.

COST AND RISK TRADE-OFF ANALYSIS OF OPTIMAL CONTROLLERS

Thesis

Submitted to the

Faculty of Miami University

in partial fulfillment of

the requirements for the degree of

Master of Science in Electrical and Computer Engineering

by

Adrianna Patch

Miami University

Oxford, Ohio

2023

Advisor: Dr. Bryan Van Scoy

Reader: Dr. David Hartup

Reader: Dr. Miao Wang

Contents

List of Tables

List of Figures

1

Introduction

The concepts of safety and cost are explored in many fields of computer science and engineering. Safety and cost are defined in a multitude of ways, both qualitatively and quantitatively. Oftentimes a control algorithm in AI or robotics operates on the basis of minimizing cost, maximizing safety, or both. To minimize cost can be considered the same as to maximize reward, and thus the ideas are often used interchangeably. Similarly, to maximize safety can be considered to be the same as to minimize risk.

In the most general sense, reward is a measurement of progress towards a goal in the given system/application. Cost is often represented as the use of resources for each unit of progress, or a measurement of decreased performance towards the goal. How resources are defined will vary between applications. Some common examples of resources are the time, energy, memory, and/or space used to achieve some level of reward. In control systems, cost in often defined in terms of the difference between the desired performance and the actual performance of a system.

In all real-world applications, safety is considered in one way or another, but this safety is not always implemented in the control process. For example, one way consumer electronics are rated for their level of protection against water and dust using the ingress protection (IP) ratings [3]. For a product to be deemed "waterproof", the second number in the IP rating must be at least 5, signifying the device withstood water jets under testing. This is a measure of safety, but it is typically not part of the control algorithm. Safety in terms of control would be more along the lines of a robotic vacuum using feedback from moisture sensors to avoid a spill on the ground, or an automated drone planning its travel based on the probability of rain.

Cost and risk relate to each other in various ways in control systems. Some control systems only optimize for one variable, whether it be purely cost or the risk associated with that cost. Sometimes the cost function itself defines the safety of the system. However, in some cases, the cost and risk defined in a system may have separate or opposing goals. Commonly in control methods that take the form of an optimization problem, one of these variables will be optimized, while the other is only vaguely accounted for.

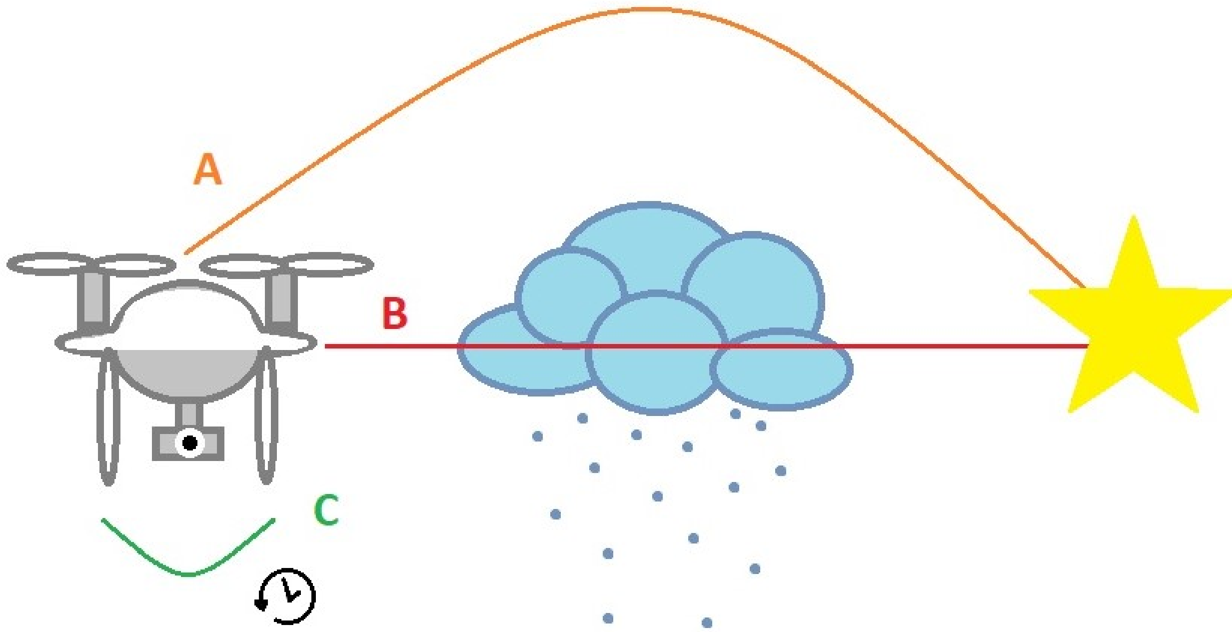


Figure 1.1: An Automated Drone Traveling in Uncertain Weather Conditions

In the example shown in Figure 1.1, an autonomous drone is traveling in the presence of uncertain weather to a goal location represented by a yellow star. Cost can be defined as the time taken to reach the goal. Risk can be defined in terms of weather damage or the likelihood of being thrown of course. Three paths are illustrated in 1.1; In Path B the drone reaches the goal with the lowest cost of time, while having the highest risk due to weather. In Path C, the drone opts to wait out the storm before attempting travel to ensure safety at a higher cost of time. Instead of purely optimizing for cost or safety, like in Path B or C, Path A accounts for some measure of both by taking a longer route attempting to avoid the worst of the weather. Depending on the relative importance of cost and safety, any of these paths could be optimal. When cost and risk have opposing goals, like in this example, it is desirable to have a controller that can trade them off for the individual needs of the system. Therein lies the motivation behind this research.

1.1 Proposed Research

In this thesis, we will examine various means of integrating cost and safety into control. We will consider existing literature in risk-aware control in the context of different system properties and applications. We will also implement and analyze the effectiveness of multiple controllers given a scenario where there is a trade-off between cost and risk, like in the example in Figure 1.1. One of said methods, will be a novel application of the Linear Quadratic Gaussian (LQG) method, which produces a controller that is optimized for a weighted combination of cost and risk. We will be referring to this method as the "Combined Cost" method.

It is the primary goal of this thesis to design this controller and demonstrate its effectiveness when compared to the other controllers implemented. To do this, we will first define a framework to quantify cost and risk in a discrete-time dynamical system. Using this problem formulation, we will then implement three existing control methods on a simple linear system. The existing methods will include a nominal LQR controller that purely accounts for cost, and then two risk-aware methods which rely on modifications to this basic nominal controller. The risk aware methods implemented are the control barrier function (CBF) seen in [4, 5], as well as the CVaR barrier function from [2]. A breakdown of these methods and necessary background information is included in Chapters 2 and 3. We will examine two scenarios in our simulations. The first scenario will a 2D robotic path planning application in the presence of obstacles. This scenario is easy to visualize and will provide great insight into the existing control methods we will be implementing. In the second scenario, we will be examining how the Combined Cost controller and the CVaR barrier function will perform in converging a robot to an equilibrium point without reaching unsafe velocities. Using the results from these simulations, we will analyze the comparative performance of the Combined Cost controller by our cost and safety metrics.

1.2 Contributions

The following items are the contributions proposed by this research:

1. A generalized problem formulation to quantify risk and cost in a given control method.
2. An optimal solution to this problem in the Linear Quadratic Gaussian setting.
3. The application of the proposed problem formulation with other controllers, and analysis of the trade-off between cost and risk in other scenarios.

1.3 Contents

The following chapters contain more context surrounding this research, as well as a breakdown of our methods and results. Chapter 2 contains background information and an overview of relevant literature. Chapter 3 consists of work completed and results. Finally, Chapter 4 concludes this research.

2

Background

In this chapter, we elaborate on concepts mentioned in Chapter 1, as well as further concepts related to our implementation of optimal control of a stochastic system. Here, we also provide a review of other methodologies that can be applied to problems of this nature, and related literature.

2.1 Control Systems Terminology

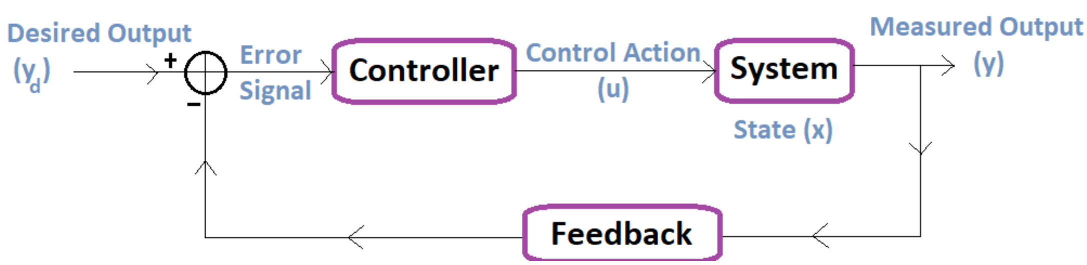


Figure 2.1: Basic Block Diagram of a Closed-Loop Control System

Figure 2.1 is a block diagram containing basic components of a control system: the controller, the system, and feedback. The system is sometimes referred to as the "plant." In robotic applications, the system is commonly referred to as the "agent." The system is the subject of control. For example, in the autonomous drone example in Chapter 1, the system would be the drone. The internal variables of a system are referred to as "state variables." In the case of the drone example, possible state variables are position or velocity. The "state" of a system refers to the values of these state variables at a given time. The "trajectory" of the system is the path of the system states as they vary with time. The controller is the algorithm used to select an input to the system which will result in a desired trajectory or end state. The input to the system is also known as the "control action" or "policy." The difference between the actual output of the system and the desired output is the "error signal." The control system in Figure 2.1 is known as a "closed-loop" system, because the controller receives feedback from the system to adaptively change the control policy based on the error signal. In an "open-loop" system, this feature is not present.

In addition to the use of these common control terms, much of our discussion will relate to the concepts of "optimal control" and "intelligent control." In the context of this research, when we say "optimal" we mean it purely in the mathematical sense. A control action is optimal in terms of a given variable, if said action is the solution to an optimization problem with said variable as the optimization variable. Similarly, when we say "intelligent control", we mean it in the sense that the control action is selected by means of artificial intelligence (AI).

2.2 System Properties

In Chapter 1 we briefly considered how factors such as cost and risk definitions, the comparative importance of each, and physical attributes of the system can all impact the effectiveness of a given controller. Certain controllers work better for different goals and scenarios. This is also true when considering particular mathematical properties of a system. The concepts of linearity, time-invariance, stochasticity, and continuity can greatly impact the complexity of a control system, and the methods available to derive a control policy. In this section, we will briefly explain each of these properties in relation to our problem.

2.2.1 Linearity

In a linear control system, the output of the system is directly proportional to the input of the system. Therefore, a linear system can simply be controlled by using some multiplier determined to achieve the desired system response. This multiplier is commonly referred to as the "gain".

State-space Representation

A system modeled by linear differential equations, can be put into a general form called the "state-space representation"

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

In an n^{th} order system with m inputs and p outputs, $x(t) \in \mathbb{R}^n$ is the vector of the system states, $\dot{x}(t) \in \mathbb{R}^n$ denotes the first derivative of $x(t)$, $u(t) \in \mathbb{R}^m$ is the input to the system, and $y(t) \in \mathbb{R}^p$ is the output of the system. A, B, C, and D, are constants describing the state, input, output, and feedthrough, respectively. The dimensions of these matrices can be seen in Table 2.1.

Matrix	Dimensions
A	$n \times n$
B	$n \times m$
C	$p \times n$
D	$p \times m$

Table 2.1: Dimensions of state space constants

Having this generalized way of representing multidimensional systems of equations is invaluable, not only because it simplifies notation, it also allows for the use of general techniques to solve and analyze the system. For nonlinear systems, it is often necessary to use methods that are fairly computationally complex. For nonlinear systems, one approach is to approximate the system by a linear system operating around an equilibrium point and then use linear control methods. However, as the system trajectory moves farther away from the equilibrium point, this model begins to lose its accuracy. In [6], which forms an optimal control method for motors with nonlinear dynamics, they use an adaptive linearization approach which is then optimized using the LQR. The adaptive approach mitigates this loss of accuracy by re-linearizing the system model around specified system operating points. Another nonlinear control method includes Model Predictive Control, which is an optimal control technique that minimizes a cost function based on the prediction of system behavior over a finite time horizon. In [7], this method is applied to a nonlinear quadcopter path planning application. The previous two methods are optimal control methods which rely on mathematical models of the system dynamics. Since nonlinear models are often more complex to form analytical solutions, intelligent control methods that make use of data and AI techniques may be preferred for nonlinear systems. In Section 2.5, we will go into further detail about optimal control methods, along with intelligent control methods.

2.2.2 Time-Invariance

In a time-invariant system, the output of a system is independent of the time at which the input signal is applied. In other words, once an input-output relationship for a system is established, it can be assumed it will remain constant. Certain methods depend on the basis of time-invariance. For example, intelligent control methods that use Markov Analysis, must assume time invariance. In the case of time-variance, tools such as the fault tree analysis, or the Bayesian network analysis may be used instead. These concepts will be explained further in Section 2.4.2. If a system is both linear and time-invariant (otherwise known as an LTI system) a large range of control methods are available for use. These methods include direct pole placement, methods involving frequency domain analysis, and the Linear Quadratic Regulator (LQR) which will be discussed more in further sections.

2.2.3 Stochasticity

A stochastic system is a system in which the trajectory is uncertain. Robustness is the ability of a system to maintain its performance despite the presence of stochastic characteristics such as noise, degradation of system components, or other disturbances. The stochastic nature of real-world systems places an emphasis on the importance of risk-aware controllers. The cost and risk directly depend on the trajectory of the system, and thus are random variables depending on the noise. For a controller to be truly robust, it must be able to account for the risk that a stochastic disturbance may undermine performance. For example, in [8], the worst-case scenario of the cost is assessed using CVaR, and used to create a control policy that will achieve the performance regardless of system stochasticity.

2.2.4 Time

Another important characteristic to consider when examining control methods is the representation of time. Continuous-time control systems and discrete-time control systems differ in how they process signals and perform computations. While a continuous-time system is often more accurate and precise, they can be more expensive to implement than discrete-time systems. When examining a problem that is already complex due to a characteristic mentioned beforehand, such as time-variance, or stochasticity, this can quickly compound into a solution that may be unfeasible for real world use. Many methods operate in discrete time for simplicity (see [2, 8–13]).

Methods used in this research can and sometimes have been used under assumptions of continuous time. For example, the CBF is often implemented in continuous time [4, 5]. This may improve the accuracy and realism in the controller, however for the topic of this research it is necessary to place limits on computational complexity where we can allow for higher quantities of data to be produced. It is predicted that continuous time systems would produce the same trends of cost and risk trade-off between the controllers examined in this study.

2.3 Cost Definitions

In the simplest terms, in a control system, cost is some measurement of the difference between the goal behavior and the actual behavior of a system. The purpose of a control system is to regulate a system to a desired behavior. Since in every control system there is a desired behavior and an actual behavior, by the previous definition every control system has a built-in definition of cost. Risk on the other hand, is not inherent to the definition of a control system. In these works [6, 14, 15], and many more, the stated control problems are completely independent of risk. This is not to say that every controller prioritizes cost. In [2,4,5,12,16,17], although cost may be accounted for, the primary objective of the system is to optimize risk.

How cost is quantified, varies with each problem. A specific example of a cost definition can be seen in [13]. When using an autonomous underwater vehicle (AUV) to observe the seafloor, the reward is the quality of data returned. To get high resolution observations, the AUV must be close to the seafloor. Therefore in this application, cost is defined as the average altitude from the seafloor.

Frequently, cost is a direct function of the error signal. For example, the cost function can simply be the absolute value of the error signal. Some commonly used cost functions can be seen in Table 2.2. The cost function used in this research takes the form of a quadratic function of the states, the inputs, along with constant matrices selected to appropriately weigh the cost of states and inputs. This form is often used in relation to LQR applications. This is discussed further in Section 2.5.1.

Cost Function	Formula
Integral Absolute Error (IAE)	$J = \int \lvert y_d - y \rvert dt$
Integral Squared Error (ISE)	$J = \int (y_d - y)^2 dt$
Integral Time Squared Error (ITSE)	$J = \int t(y_d - y)^2 dt$
Mean Squared Error (MSE)	$J = \mathbf{E}[(y_d - y)^2]$
Linear Quadratic (LQ)	$J = \int [x^\top Qx + u^\top Ru + 2x^\top Su]dt$

Table 2.2: Common Cost Definitions

2.4 Safety Definitions

As mentioned before, the concept of safety is not inherent in the definition of a control system. Since the concept of safety in control systems is so vague, safety functions do not have a common root for definition as cost functions do with the error signal. The ways in which safety are defined can be drastically different from each other, depending on the problem definition and application. In this section, we delve into different definitions of safety in the context of control systems.

2.4.1 Safety Critical

Perhaps the simplest view of safety is the Safety Critical definition. By this definition, a state is either safe or unsafe, with no ability to represent partial safety. What determines the state's safety can be predefined or can be determined by constraint functions. One safety critical example is [12], which plans the path of a heterogeneous team of robots in the presence of obstacles. In this example, the environment was represented in discrete tiles, where each location is a potential state. The unsafe states were predefined as tiles that contain obstacles. In a more complicated safety critical example, safety may depend on numerous variables. For example, [1] illustrates the interdependencies between events that contribute to an overall event of failure. The diagram representing their example is pictured in Figure 2.2.

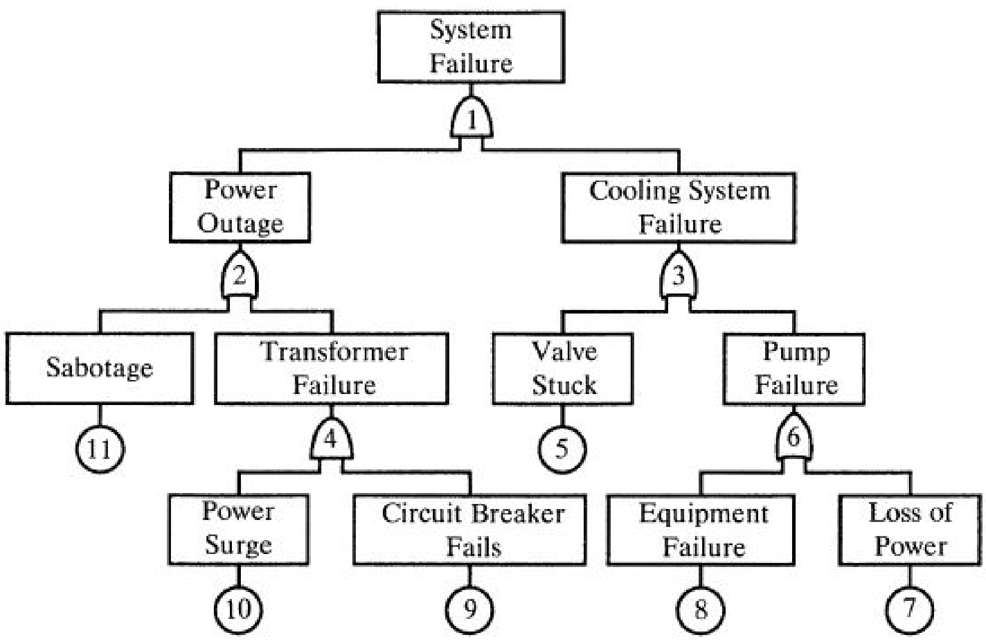


Figure 2.2: System with multivariable dependencies, illustrated in a fault tree. Figure source: [1].

Safety critical controllers often rely on constraint functions. Two common tools in developing constraints are the Control Lyapunov Function (CLF) and the Control Barrier Function (CBF).

Control Lyapunov Function (CLF)

The control Lyapunov function relies on the concept of Lyapunov stability. By definition, a state is Lyapunov stable if said state remains near the equilibrium point it started near for all time. More restrictively, a state is asymptotically stable if said equilibrium point is zero. One way to define safety is through this stability. In other words, a system is safe if its states are stable. The control Lyapunov function technique finds a controller, u, which guarantees stability. To explain this process, we must first define the Lyapunov function $V(x)$, which is a positive definite function that is radially unbounded. This means that $V(x)$ is continuous, differentiable, and as $V(x)$ approaches infinity, the magnitude of the state, x, grows without bound. It can be challenging to find a suitable Lyapunov function, but once found it is a valuable method to ensure stability in a system. A common choice of Lyapunov function for the CLF technique is a quadratic function in the form $V(x) = x^\top P x$, where P is a positive definite matrix appropriate for the given system. If $V(x)$ is found, and the control policy can ensure that $dV(x)/dt$ is negative definite, then the system is guaranteed to be globally asymptotically stable. A common implementation of control using CLF is to form an optimization problem to choose a controller, $u(x)$, as a function of x that minimizes or maximizes $dV(x)/dt$ subject to some constraints.

Control Barrier Function (CBF)

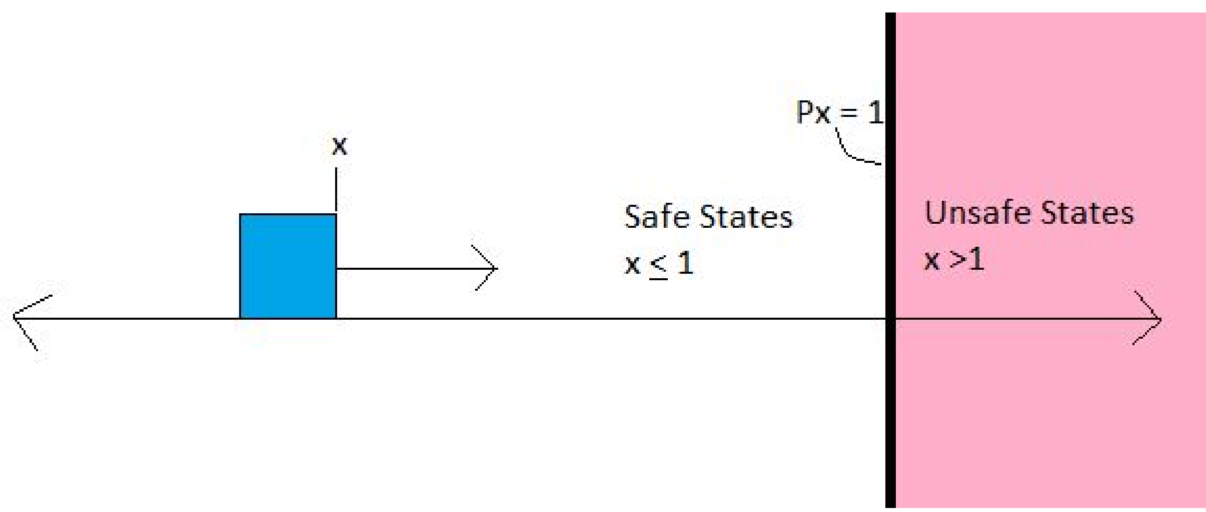


Figure 2.3: Illustration of barrier function example with an agent traveling along a trajectory in the presence of a wall.

Another tool used in control theory is the Control Barrier Function (CBF). While the CLF is designed purely to guarantee the stability of a system, the CBF is designed to keep the system states within a predefined safe set. This is useful in systems that have safety constraints other than stability. Suppose a system has a clearly defined safe set S. The barrier function, $h(x)$, is a mathematical representation of the border around S. In other words, $h(x) \geq 0$ when x is within S, otherwise $h(x) < 0$. Consider a robot traveling along a trajectory, as in Figure 2.3. If there is a wall at position $p_x = 1$, the barrier function could be defined as

$$h(x) = p_x - x \geq 0.$$

Any attempt to cross the wall into the unsafe set would violate this constraint.

Similar to the CLF, the CBF can be used in tangent with optimization techniques to produce a control law that meets this constraint. Typically, this constraint takes the form of a function that ensures the trajectory of the state remains some distance away from the current distance to the barrier by some proportion α,

$$h(x_{t+1}) \geq \alpha h(x_t) \geq 0.$$

The control barrier function is sometimes used in combination with the control Lyapunov function to ensure safety based on both the defined constraints and stability. This can be seen in [4, 5, 16, 17].

To conclude this section, safety critical control is appropriate for cases where there are clear distinctions between what is safe and what is not safe. It is also a valuable way to set safety as the number one priority in situations where there is no room for risk allowances. For example, if the violation of a safety constraint would result in disastrous failure or injury to workers/consumers, a safety critical approach may be preferred. However, in cases where violation of a system constraint has less drastic consequences, or an extremely low likelihood of occurring, it may be useful to define safety in a less binary way. For example, safety can be represented in discrete levels, or as a probability of an outcome.

2.4.2 Probabilistic Safety

In probabilistic safety, safety is characterized as the probability of some event. It is important to note that although safety critical control can be defined using probability, when we refer to probabilistic safety, we are referring to cases in which safety is represented by a probability. For example, a safety critical definition could be that a system is safe if the probability of an event occurring is below some threshold. While in a probabilistic safety definition, the safety itself is a probability. Using this definition, instead of referring to a system as either safe or unsafe, we can say the likelihood of safety is some percent value.

In stochastic, risk-aware systems, many safety definitions are formed on the basis of probability. Our own problem formation proposes a probabilistic safety definition. Our definition of risk will take the form of a cumulative probability of the risk at every stage along the system trajectory given some application specific definition of risk at every stage. The particular approach to doing this is outlined in the Completed Work, Section 3.1. Our safety definition requires that the complete trajectory of the system be known, and for probability of failure at each stage to be independent across time. Since our goal in defining safety this way is simply to quantify risk for the sake of comparison amongst controllers, it can be assumed the trajectory will be known. However, many safety definitions in the literature are formed on the basis of incomplete knowledge or even dependencies between the probabilities of each state. The following sections detail other probabilistic safety methods found in the literature.

Fault Tree Analysis

Fault Tree Analysis is a logic tree based method in which the nodes are events, and the higher level nodes are the unions or intersections of these events. This is referred to as a "top-up approach" because it begins at the root node, or the overall event of failure, and backtracks to analyze the relationship of events that directly contribute to it. Given this structure, one can recursively form the equation of the root node that determines the overall system safety. A visual example of this method was included previously in Figure 2.2. This method can be safety critical or probabilistic. In the safety critical example illustrated in Figure 2.2, [1] details an example in which the relationships between binary events in a fault tree can be modeled with logic gates. The probability of the root node can then be directly simplified to a Boolean equation. If the events are probabilistic instead of binary, the relationship between events can be represented with unions, intersections, and complements. This is a way to predict the probability of the overall system failure, and thus a valuable tool for safety analysis. This literature, [18], used fault tree analysis to map probabilities to various accidents that may occur in a system. These accident event probabilities are then weighed by severity and combined in an overall safety value assessment.

Markov Analysis

Another probabilistic modelling technique is Markov Analysis. This method makes use of the transition probabilities between states of a system. In this technique, all the possible states of a system are identified, including unsafe states. Consider a system with n possible states. P is a row vector of length n representing the probability of each state occurring.

If a system is known with absolute certainty to be in state n, states $1, 2, ..., n-1$ will have a probability of 0, and state n will have a probability of 1. So $P = [0, 0, ..., 1]$. Markov Analysis uses the following relationship:

$$P_t T = P_{t+1}$$

where T is an $n \times n$ matrix in which the rows and columns correspond to each possible state. Each cell contains the probability of transitioning between the given row and column state. A Markov Chain is when this analysis is repeated for some number of time steps to produce the expected probability in each state at the end of the specified time duration. Since the unsafe states are predefined and Markov Analysis returns the probability of each state occurring in the future, safety can be defined as the probability of any of the unsafe states occurring.

The following works use some form of Markov Analysis to define safety in their problem formation: [18–20]. In a safety critical example, [21] uses Markov analysis to determine some probability of reaching unsafe states, and defines safety as this result being below some threshold value. While Markov Analysis proposes a simple problem formation, it can be challenging to accurately assess the probability of moving from one state to another in real world systems. This can also be complex if there are a large number of possible states. Markov Analysis also relies on states being independent of one another, and for the system to be memoryless and time-invariant.

Bayesian Network Analysis

In stochastic systems where there are interdependencies between variables, Bayesian Network Analysis may be a better approach than Markov Analysis. Similarly to Fault Tree Analysis, Bayesian Network Analysis models the relationships between events related to the safety of a system in the structure of a diagram. However, while Fault Tree Analysis uses a top-down approach, Bayesian Network Analysis takes a bottom-up approach. This means that instead of beginning with the specific event defining safety, Bayesian Network Analysis typical begins with prior and current system knowledge of states and builds a network from the ground up. This Bayesian Network, or “belief” network is modeled in the form of a directed acyclic graph, where the nodes are the states of the system and edges direct each state to the possible states they may lead to. This approach is used for causal systems, unlike Markov Analysis, which can only be used for memoryless systems. The probability of each state is determined by the probability of previous states using Bayes’ theorem and partial knowledge of the conditional probabilities in a system:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}.$$

Bayesian Network Analysis is a tool commonly used in artificial intelligence approaches to control. Reference [22] uses Bayesian Networks in their quantitative approach to analyzing the safety of unmanned aerial vehicles.

2.4.3 Value at Risk (VaR)

Another definition of safety is Value at Risk (VaR). This technique was actually developed for financial risk management, but has since been applied to other industries in which statistics are used, including controls engineering. The benefit of VaR is that it provides a statistical measure of the worst-case scenario.

In the context of safety analysis, the process used to calculate the VaR goes as follows. Consider a stochastic system with output $F(x, u, w)$ where x, u, and w are the state, input, and process noise. The states of this system are then mapped to a range of values that determine the desirability of said state using some continuous function $h(x)$. Lower values of $h(x)$ correspond to less desirable outcomes, in this case riskier outcomes, and higher values signify safer outcomes. On this scale from least desirable to most desirable, there is a threshold value that determines whether an outcome is safe or not. Since this is a stochastic system, the evaluation of this function will have some distribution of possible outcomes. Due to the uncertainty in the system, a single state and control input can produce outcomes both below and above the threshold for safety.

One approach to this problem would be to select a control input that guarantees that the lowest possible value of h is above the safety threshold. While this would guarantee safety given there is a control input that will satisfy this condition, it can be quite expensive and inefficient. Consider the case where the optimal control input produces a distribution of h where there is an outlier below the safety threshold. While this outlier poses very little risk to the safety of the system, in this approach a sub-optimal control input must be selected to satisfy the harsh safety constraints.

Another approach is to use the expected value of the h distribution as the means to measure the expected performance against the safety threshold. This approach is much less strict than the previous, but depending on the distribution of h it can be misleading. For example, if there is a large portion of the distribution above the threshold, even if there is a nonnegligible portion below the threshold, the system would still be considered safe.

In the VaR approach, the performance measurement depends on a given confidence level $\beta \in [0, 1]$. The VaR is found by defining the tail end of $P(h)$, which is the bottom $\beta\%$ of the probability distribution of h. The VaR is the highest value of h in the tail end, in other words the best-case scenario of the worst expected outcomes. If the VaR is above the defined safety threshold, the system is considered "VaR safe" for the specified state, input, and noise level.

One way the confidence level can be selected is to specify some allowable level of uncertainty in the safety analysis. If past data or current system knowledge suggests that extremely undesirable outcomes are rare, it would suggest that even for a low confidence level VaR can guarantee safety. In some industries, such as banking, the confidence level of VaR may be mandated by regulation. According to the Legal Information Institute [23], a Board-regulated institution must use a 99.0% confidence level for its VaR-based measurement.

Conditional Value at Risk (CVaR)

There have been several modifications of VaR used in controls engineering methods, including CVaR, which is a key component of one of the implementations of in this research. CVaR takes the concept of VaR safety one step further. While VaR is the value of h at the upper end of the tail end of the distribution. CVaR is the expected value of the tail end of the distribution. Reference [2] provides a graphical example of a distribution where given β, the VaR, and the expected value of $p(h)$ is safe, CVaR safety is not satisfied. The figure from [2] is shown in Figure 2.4. Other controls literature making use of CVaR safety include [8,11,24–26]

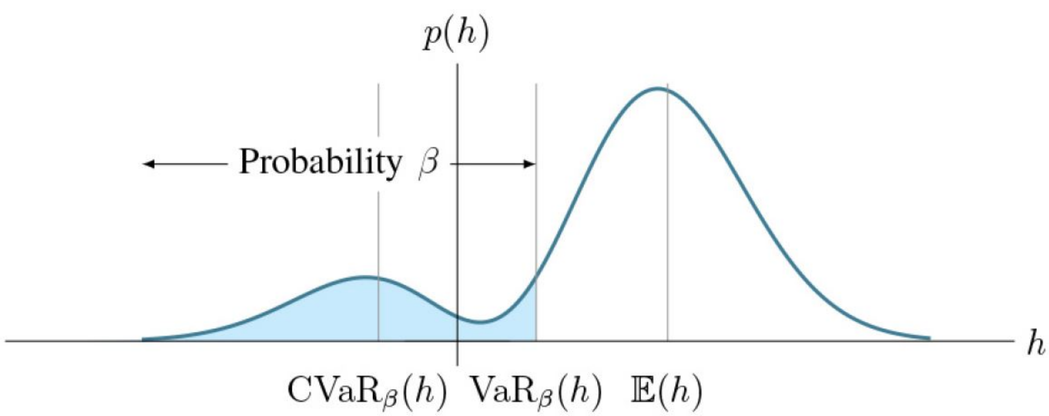


Figure 2.4: Example of the expected value, VaR, and CVaR of a probability distribution with confidence level β. Figure source: [2].

2.5 Methods

There are many control methods that integrate the cost and safety definitions outlined in the previous section. When reviewing literature related to cost and safety in control systems, we found that a large portion of risk-averse control methods can be categorized under optimal control or intelligent control. These types of control systems are robust, making them suited for applications involving risk. However, both methods have limitations that prove more challenging for certain applications. A fundamental difference between intelligent control and optimal control is that optimal control relies on a predefined model of the system, while intelligent control uses AI techniques to learn a control policy with the direct use of the system data.

2.5.1 Optimal Control

As mentioned before, the optimal control problem requires a mathematical model of the system dynamics. The model typically takes the form of a set of differential equations that map the state of a system to a control input, among other variables. For linear systems of differential equations, optimal control methods make use of the state-space representation defined in Section 2.2.1.

In optimal control, given a dynamical system model, an optimal control strategy is formed to minimize some objective function while adhering to possible system constraints. In the context of a risk aware controller, the objective function is usually some convex function of cost, and then constraints are selected based on some definition of safety. This can be an open-loop strategy where all control inputs are calculated ahead of system operation, or a closed-loop system that adapts the control inputs in real time. Some popular optimal control methods include Model Predictive Control, the Linear Quadratic Regulator/Gaussian, and optimization problems using tools like the CBF/CLF to form constraints.

Linear Quadratic Regulator (LQR)

The Linear Quadratic Regulator (LQR) is an example of a closed-loop feedback optimal controller. As implied by the name, the LQR assumes the system dynamics are linear, $F(x_t, u_t) = Ax_t + Bu_t$, and the function to be optimized is quadratic. In the context of control systems, the LQR is applied to a cost function depending on the system states x and the control action u. Below is the general equation for the stage cost at time t:

$$J(x_t, u_t) = \begin{bmatrix} x_t \\ u_t \end{bmatrix}^{\intercal} \begin{bmatrix} Q & S \\ S^{\intercal} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix}.$$

The LQR uses the following constants as a way to weigh particular states and inputs: the state cost $Q \in \mathbb{R}^{n \times n}$, the input cost $R \in \mathbb{R}^{m \times m}$, and the cross-correlation matrix $S \in \mathbb{R}^{n \times m}$ which is null in many applications. Given the stage cost, the overall cost is J, where

$$J = \sum_t (x_t^\top Q x_t + u_t^\top R u_t + 2 x_t^\top S u_t).$$

The objective is to find the gain K associated with the state feedback control law $u_t = -K x_t$, which minimizes J.

In the LQR method, the solution to this problem is

$$K = (R + B^\top P B)^{-1} (B^\top P A + S^\top),$$

where P is the solution to the discrete-time Riccati equation, which is also seen in [27]:

$$P = A^\top P A - (A^\top P B)(R + B^\top P B)^{-1}(B^\top P A) + Q.$$

For a stochastic system with Gaussian process noise w, LQR is used in combination with a Kalman filter, which is to provide an estimate of the stochastic state. This is called the Linear Quadratic Gaussian (LQG). The LQG applies to systems of the form $F(x_t, u_t, w_t) = Ax_t + Bu_t + w_t$, and calculates gain in the same way as the LQR controller. The only difference is that it relies on state estimations from the Kalman filter to obtain knowledge of the trajectory of the system. This is the nominal controller used for our implemented methods in Section 3.1.2, as well as the basis for the Linear Exponential Quadratic Gaussian (LEQG).

Linear Exponential Quadratic Gaussian (LEQG)

The LEQG is an extension of the LQG control problem. When using state estimator to assume the trajectory of the system, there will be uncertainty within the system. The LEQG is designed to compensate for this randomness. This is particularly useful when risk is the optimization variable due to its probabilistic nature. This is true in the case of our problem formation outlined in Section 3.1.

Similarly to the LQG, the LEQG controller is solved using a Riccati equation. The key difference in LEQG is that the form of the cost function is the exponential of a quadratic which makes use of a risk-sensitivity parameter, θ. This parameter dictates whether the function used to quantify safety will be minimized or maximized with respect to the control policy. If $\theta < 0$ the safety function will be minimized under the optimization problem. In other words, the controller that optimizes the objective will be risk-seeking. While $\theta > 0$ corresponds to a risk-averse controller, where the function will be maximized. When $\theta = 0$, the controller will be risk-neutral, in which case the function will be reduced to the standard expectation. The general form of this function is defined in [28] as:

$$X_\pi = -\theta^{-1} \log[E_\pi(e^{-\theta C})],$$

where $E_\pi(\cdot)$ is the expected value with regard to the control policy, π, and C is a cost function in the Linear Quadratic form.

2.5.2 Intelligent Control

As opposed to optimal control which relies on dynamical modeling, in intelligent control the problem is mapped to some AI technique, such as fuzzy logic, rule-based systems, or machine learning. A benefit of optimal control is that it provides a precise measure of the optimal performance of a system and can have great success in influencing the system's states towards that behavior. However, in some cases a mathematical model can be too complex, resulting in computationally expensive solutions. Furthermore, in cases of high uncertainty or incomplete knowledge of the system, it may not be feasible to arrive at an accurate mathematical model in the first place. In situations like these, intelligent control may be a better approach. In some literature, hybrids between intelligent and optimal control methods are applied to seek benefits from both approaches. For example [15] uses concepts from fuzzy logic in a model predictive control method. Many of the methods using the tools outlined in Section 2.3.2 use AI applications. A notable example is Markov Analysis applied to Markov Decision Processes (MDPs). MDPs, or Partially Observable MDPS (POMDPS), are popular decision-making frameworks used in intelligent control methods based on reinforcement learning.

2.5.3 Trading off Cost and Risk

Thus far, we have examined numerous ways of defining cost and risk, as well as the overarching approaches of optimal vs. intelligent control, in the context of different scenarios and system properties. Another component of risk-aware control that is of particular relevance to our problem is the manner in which cost and risk are balanced. In this section we will look at a few ways that the relationship between cost and risk manifests in the literature.

Tuning Parameters

A common feature of control methods we have examined are the presence of tunable parameters that directly affect cost and risk. In optimal control methods, a common problem structure is one in which a utility function is being optimized for cost, subject to some constraint that accounts for risk. In these types of problems, the risk constraint may be tightened or loosened. The following literature features methods resembling the aforementioned problem structure: [2, 4, 26, 29]. One of the methods in [4] has a min norm based cost utility function, with both CBF and CLF constraints. The CLF constraint in this method contains a "relaxation variable" which can be adjusted to trade off performance between cost and safety.

In [2] cost is optimized using the LQR control method, subject to a CVaR safe CBF-based constraint. They differentiate between different levels of risk-aversion by varying the confidence level of the CVaR function demonstrating a clear trade-off between cost and safety in their simulated results. The differentiation between levels of risk by manipulating a CVaR-based safety constraint can also be seen in [24] with the "risk-aversion level", [9, 26] with "risk tolerance" parameters. In the previously mentioned LEQG method [28], we have the "risk-sensitivity" parameter which fulfills the same purpose.

Direct Mediation Between Cost and Risk

While the previous methods all contained some tunability to trade off cost and risk, they do not offer a variable to directly mediate between the two goals. Having such a variable is one of the primary features of the Combined Cost method proposed in this research. In this section we will look at a few other methods that use a variable to weigh cost against risk in their control algorithm.

The first example in the literature where we see this is in this autonomous safety management model proposed in [18]. This is an intelligent control method that uses a POMDP to model safety states of the system, and maps these states into the likelihood of accident using Fault Tree Analysis. By doing this, given a control action, a risk assessment is determined taking into account both the probability and severity of possible accidents resulting from said action. Their model determines a total utility function which is a weighted sum of safety (V_{Safe})

and reward (V_{Task}). The utility is a function of the control action a and is denoted by:

$$V_{Tot}(a) = W_s V_{Safe}(a) + (1 - W_s) V_{Task}(a),$$

where W_s is the weighting parameter that provides the relative importance between the two goals. The control action is then selected to maximize V_{Tot}.

Another example is this intelligent control method in [30], where the goal is to trade off “maintenance cost” and “failure cost” in an intrusion detection system. In this scenario the failure cost can be interpreted as the risk parameter. A POMDP is used to model the state space. The model accounts for many factors related to safety, including human factors such as the knowledge and motivation of a potential attacker of the system. The model was validated using real intrusion detection scenario datasets. They used an overall cost function containing a variable $\beta_i \in [0, 1]$ to balance the trade-off between maintenance cost and failure cost. The desired rational response, or control action, was selected to minimize this cost function.

Another intelligent control method that uses a weighting parameter to trade off risk and reward is this reinforcement learning method [31]. This work proposes a Reward-Punishment Actor-Critic algorithm in a problem of robotic manipulation of objects. This takes inspiration from the biological framework of the reward and punishment inside the brain of animals that motivates them in their decision-making. A robotic manipulator is covered in sensors, they call “skin cells”, which provide tactile information to the control algorithm. This aids in the goal of the manipulator, which is to push an object to a goal location. In this example reward is a function of the overall distance to the goal location, and the punishment (risk) function penalizes the manipulator for losing contact with the object at the risk of losing it. While both the reward and punishment functions contain tunable parameters to individual weigh each of these functions, the overall utility function also incorporates a weighting variable w. The variable w is defined as the ratio of reward and punishment. In the overall utility function w is a coefficient of terms related to reward, and its complement is a coefficient of punishment terms, effectively enforcing a balance of risk and reward.

3

Completed Work

This chapter includes the detailed contributions of our work. First, we outline our general problem formation. Our proposed problem formation provides an intuitive framework that can be used to quantify risk and reward for many control problems. Additionally, it lays a crucial foundation for the comparison between the Combined Cost method and existing methods. Using these cost and safety definitions, we compare some existing control techniques through simulation in a two-dimensional robotic path-planning problem. Finally, we provide a novel way of using the LQG control method to trade off cost and risk. We implement this method along with previously examined methods on a simple one-dimensional double integrator system to demonstrate the comparative performance of our method in terms of minimizing cost and risk.

3.1 General Problem Formation

For the following definition of our problem formation, we will assume a stochastic, linear, discrete-time system, in the form

$$x_{t+1} = F(x_t, u_t, w_t)$$

where x_t, u_t, and w_t are the state, control input, and process noise respectively at time t. Since we are examining the system in the state-space representation, and operating on the assumption that we can measure the state, we can presume some state-feedback controller. For the purpose of this general framework, we will denote it as

$$u_t = K(x_t).$$

3.1.1 Cost

At every time, t, there is an associated stage cost. This stage cost is a function of the state and the control input, and is denoted by $c_t(x_t, u_t)$. The total cost of the controller, K, can then be defined as the cumulative sum of each stage cost in the trajectory of the controller:

$$\text{cost}(K) = \sum_t c_t(x_t, u_t).$$

The stage cost, $c_t(x_t, u_t)$, can be defined in a variety of ways. This includes methods mentioned in Section 2.2. For the purpose of our specific implementations, we are assuming a quadratic cost function:

$$c_t(x_t, u_t) = \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T Q_t^{cost} \begin{bmatrix} x \\ u \end{bmatrix} + (R_t^{cost})^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + S_t^{cost},$$

much like the cost function defined in Section 2.3.

3.1.2 Risk

Similar to cost, at each stage there is an associated risk that is a function of the state and the control input, which we denote by $r_t(x_t, u_t)$. If f_t is a Boolean random variable representing the probability of failure at each stage, then $r_t(x_t, u_t) = p(f_t \mid x_t, u_t)$. The total probability of failure is then the probability of failure at any stage of the trajectory, given the state and control input at each stage. In other words, the total risk of the controller is the union of the probability of failure at each stage. If there are n stages in the trajectory, T, this union can be written as

$$p(f_{t_1} \cup f_{t_2} \cup ..., f_{t_n} \mid T).$$

The opposite of risk is the complement of risk, or the probability of success:

$$1 - \text{risk}(K) = p(\neg(f_{t_1} \cup f_{t_2} \cup ..., f_{t_n} \mid T)).$$

Using De Morgan's law,

$$p(\neg(f_{t_1} \cup f_{t_2} \cup ..., f_{t_n} \mid T)) = p(\neg f_{t_1} \cap \neg f_{t_2} \cap ..., \neg f_{t_n} \mid T),$$

where the probability of not failing is denoted as $p(\neg f_t) = 1 - p(f_t)$.

When looking at the complement of failure events across time, we can assume conditional independence. This means that for any two independent times,

$$p(\neg f_{t_1} \cap \neg f_{t_2} \mid T) = p(\neg f_{t_1} \mid T)p(\neg f_{t_2} \mid T).$$

This goes to show that the probability of not failing at an individual stage of the trajectory only depends on the state and control action at the specific time step of the stage in question. In other words, $p(\neg f_t \mid T)$ can be reduced to $p(\neg f_t \mid x_t, u_t)$.

With this separation of dependencies, we can then represent the overall probability of not failing along the entire trajectory as a cumulative product of the probability of not failing at each stage:

$$\prod_t p(\neg f_t \mid x_t, u_t).$$

Recall that the complement of the probability of not failing at any stage is the probability of failing at that stage. Thus, it follows that

$$\text{risk}(K) = 1 - \prod_t (1 - r_t(x_t, u_t)). \tag{3.1}$$

Notice that given this definition, if any stage risk has a 100% chance of failure, then $r_t(x_t, u_t) = 1$. Therefore risk(K) evaluates to 1, guaranteeing failure. The probability of failure at each stage can be designed to suit the requirements and specifications of the given system. In our implementations, we will define stage risk per the control scenario.

3.2 Control Implementations in two Scenarios

Having a generalized definition of cost and risk, we were able to implement this idea on some existing control method and our proposed method using MATLAB. One existing method is based on the CBF, and another is a modified version of the CBF method that guarantees CVaR safety. In analyzing the aforementioned control methods in this thesis, we have conducted implementations in two scenarios. The first scenario is a 2D robotic path planning application in the presence of obstacles. This scenario is useful in the visualization of risk and reward, as well as the comparison between the existing control methods examined in this research. The second method is a 1D scenario meeting the LQG criteria required for the proposed method in this research. This scenario is less tangible to visualize, but provides a ground of comparison between the existing methods and the Combined Cost method.

3.2.1 Scenario 1: 2D Path Planning

While the CBF and the CVaR barrier function can be applied to many systems, a practical example is robotic path planning in an environment with obstacles. This is a common example application of the CBF, not only because of the expanding use of robotics, but partly because it is intuitive to have a barrier function corresponding to the physical boundary of an obstacle. For example, in Figure 3.1 the obstacle is circular, so the physical boundary of

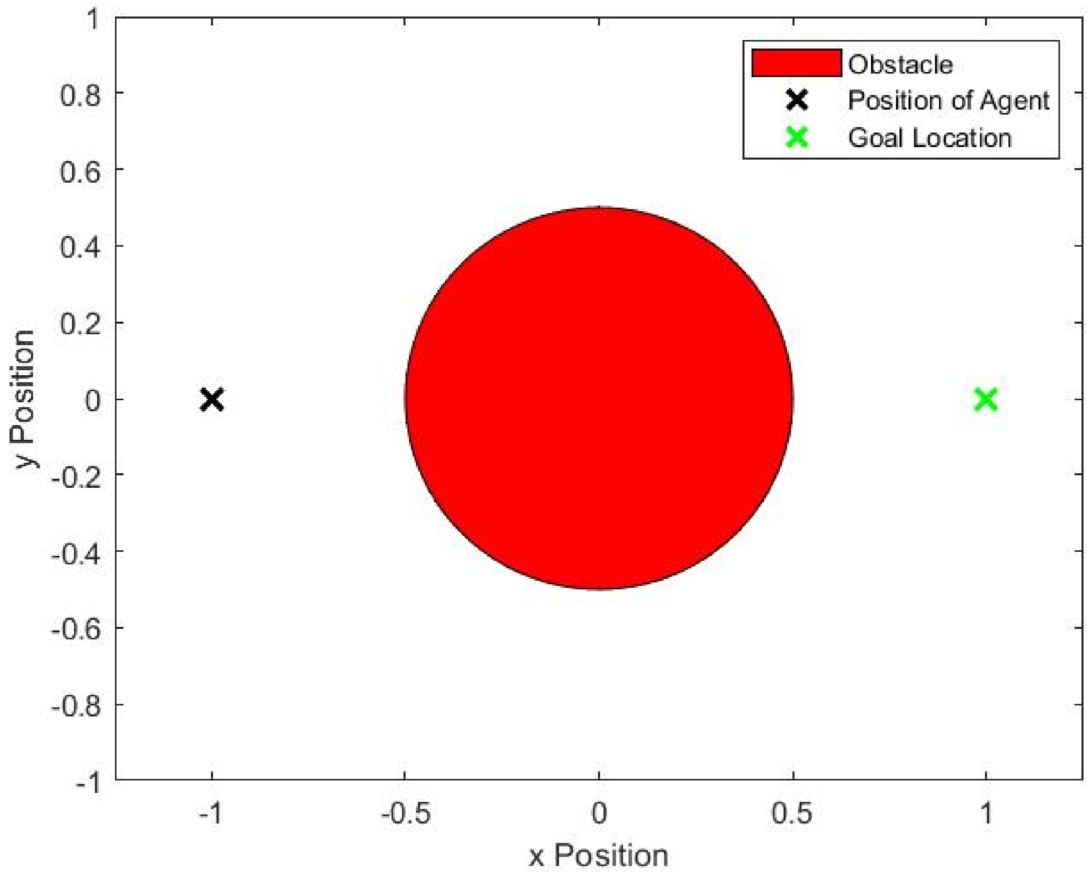


Figure 3.1: Example of a path planning scenario with an obstacle

the obstacle can be represented by the equation of a circle.

System Dynamics

While the dynamics of a robot can be quite complex, a simple model for a planar mobile robot can be modelled as a simple linear double integrator system where the control input $u(t)$ is the acceleration of the agent, and the output $y(t)$ is the position $x(t)$ of the agent.

The system dynamics are as follows:

$$\frac{d^2x}{dt^2} = u(t),$$

$$y(t) = x(t).$$

Assuming a two-dimensional coordinate system, we used the following discrete-time state space model with sampling time t_s to represent our system dynamics:

$$F(x_t, u_t, w_t) = \begin{bmatrix} 1 & 0 & t_s & 0 \\ 0 & 1 & 0 & t_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} \frac{1}{2}t_s^2 & 0 \\ 0 & \frac{1}{2}t_s^2 \\ t_s & 0 \\ 0 & t_s \end{bmatrix} u_t + w_t$$

where w_t is the process noise, assumed to follow a Gaussian distribution. Here, the state of the system is

$$x = \begin{bmatrix} p_x \\ p_y \\ v_x \\ v_y \end{bmatrix} \qquad \text{where} \qquad \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \dot{p_x} \\ \dot{p_y} \end{bmatrix}.$$

The states p_x, p_y, v_x, v_y are the x position, y position, velocity in the x direction, and velocity in the y direction respectively.

Nominal Controller

In this path planning application, the objective is to minimize the difference between the current state of the system x_c, and a reference signal x_r, while adhering to some constraints. Both the CBF and the CVaR barrier function methods operate on the basis of taking some nominal controller, and modifying it to suit the constraints of the system. In these implementations, the LQR is used to find a nominal controller that will return the optimal trajectory from the initial state x_0 to the goal state x_r. This will find the cost-optimal path in the absence of obstacles. The design parameters used for the development of the controller are as follows. The state cost Q, is $50 \times \mathbb{I} \in \mathbb{R}^{n \times n}$. The input cost R is $.001 \times \mathbb{I} \in \mathbb{R}^{m \times m}$. The initial state covariance S, is assumed to be 0. Following the process outlined in Section 2.5.1, we calculate the gain k resulting in a nominal controller in the form $u_{nom} = k(x_c - x^r)$.

CBF

As defined in Section 2.4.1, the CBF is a tool in control systems that uses some barrier function, $h(x)$, to describe the boundary between safe and unsafe states in a given system. In the case of the environment in Figure 3.1, there is a single obstacle in the form of a circle centered at $(0, 0)$ with radius 0.5. Recalling the definition of our state variables, x_t will denote the state vector at time t. p_x^t and p_y^t will denote the first and second state variables representing the x and y positions at time t. So for this scenario, the barrier function in Figure 3.1 can be written as

$$h(x_t) = \begin{bmatrix} p_x^t \\ p_y^t \end{bmatrix}^\top \begin{bmatrix} p_x^t \\ p_y^t \end{bmatrix} - (0.5)^2.$$

This implementation of the CBF is an optimal control method that can be represented by the following optimization problem:

$$\begin{aligned} \min_u \quad & ||u_t - u_{t,nom}||^2 \\ \text{s.t.} \quad & \underline{u} \leq u_t \leq \overline{u}, \\ & h(F(x_t, u_t)) \geq \alpha h(x_t), \end{aligned} \tag{3.2}$$

where $\underline{u}$ and $\overline{u}$ are the upper and lower bounds to the control input, and $\alpha > 0$ is a tunable parameter used to set the stringency of the control constraint. This optimization problem aims to find the control input closest to the input provided by the nominal controller, all while ensuring that the barrier function of the resulting output $h(F(x_t, u_t))$ remains α times as far from violation as the barrier function of the current state $h(x_t)$. Although $F(x_t, u_t)$ may be dependent on noise (w) as well as x_t and u_t, the CBF implementation is not equip to account for noise. It is also worth noting that for cases where there are unsafe regions, each requiring unique barrier functions, multiple constraints may be added to the optimization problem.

CVaR Barrier Function

The CVaR barrier function follows the same process as the CBF method, with the exception of the safety constraint. In Section 2.3.3 we defined CVaR safety as the expected value of the tail end of some distribution of outcomes, where the tail end represents the left-most portion of the probability distribution. The percentage of outcomes included in the tail end is defined by a given confidence level β. Below is the optimization problem in terms of the

CVaR constraint, where $CVaR(h)$ returns the CVaR of the barrier function:

$$
\begin{aligned}
\min_{u} \quad & ||u_t - u_{t,nom}||^2 \\
\text{s.t.} \quad & \underline{u} \leq u_t \leq \overline{u}, \\
& \text{CVaR}(h(F(x_t, u_t, w))) \geq \alpha h(x_t)
\end{aligned} \tag{3.3}
$$

This modified constraint ensures that the worst expected outcome remains α times as far from violation as the barrier function of the current state $h(x_t)$. In [2], CVaR(h) is defined by the following optimization problem:

$$
\text{CVaR}(h_t) := \inf_{\zeta \in \mathbb{R}} \mathbf{E} \left[\zeta + \frac{(-h_t - \zeta)_+}{\beta} \right]
$$

In this definition, the optimization variable ζ is the CVaR. Note that $(\cdot)_+ = \max\{\cdot, 0\}$. This function is implemented in MATLAB using the `fminunc` solver, while the overarching optimization problem for this implementation and the CBF implementation were solved using `fmincon`.

Stage Risk

To calculate risk using the definition in our problem formation, we first had to define stage safety. In the calculation of stage safety, we related the probability of failure at each time step to the relative proximity to the obstacle. Given some buffer distance d away from the barrier function, we form a gradient of safety from the center of the obstacle to the buffer, where the center point of the obstacle corresponds to 0, and $x \geq h(x) + d$ corresponds to 1. We developed this function to characterize the stage cost in an environment with a single circular barrier with radius rad centered at the origin:

$$
r_t = 1 - s_t \qquad \text{where} \qquad s_t(h_t(x_t, u_t))) = \min\left(\frac{h_t + rad^2}{d + rad^2}, 1 \right).
$$

While this circular barrier is convenient for this demonstration of this method, any shaped barrier/barriers can work. One way to implement arbitrary barrier functions could be to include a random list of polygon functions as constraints. Depending on the size and locations of these barriers and the strictness of the control parameters, reaching the goal location may or may not be feasible.

Simulation

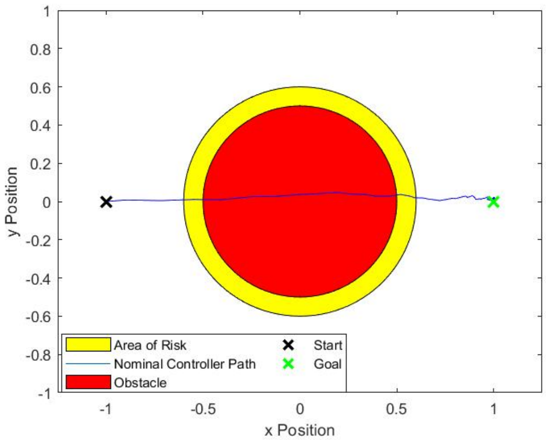
Both of the methods described were implemented on the double-integrator system dynamics in the basic environment with a circular barrier with initial and goal positions on either side. The following initial and goal states were used in the simulation:

$$x_0 = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{and} \qquad x_r = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

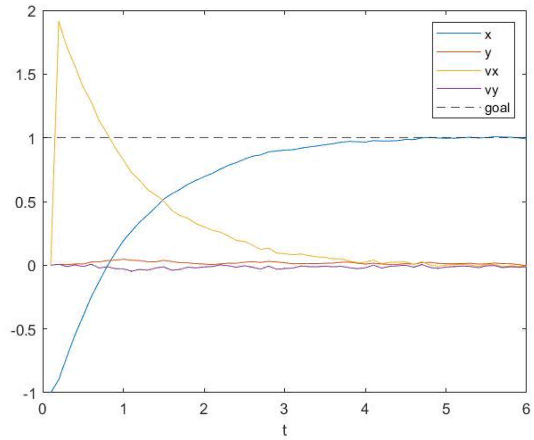
We simulated Gaussian noise with the covariances 3×10^{-5}, 5×10^{-5}, 3×10^{-5}, and 5×10^{-5} for process noise in states p_x, p_y, v_x, and v_y respectively. To give some perspective on how large this noise is, we looked at how much of the change in position and velocity at each time step is noise on average. For the nominal controller trajectory, on average the signal-to-noise ratio for the change in position and velocity at each time step was around $2.2, .95, 7.1$, and 1.8 for the states p_x, p_y, v_x, and v_y. For example, this would mean that for every change in position, p_x, the change resulting from the signal is 2.2 times stronger than the noise. The noise in the y direction is much stronger, due to not only the comparatively higher noise covariances, but the fact that the nominal controller takes a horizontal path. Figures 3.2, 3.3, and 3.4 show an example run of the simulation for the nominal, CBF, and CVaR controllers. The parameters for stringency and the confidence level were set to $\alpha = 0.85$ and $\beta = 0.1$. These values correspond to a risk averse setting with a fairly high stringency in the constraints and a low confidence level. For our stage risk calculation, we used a buffer distance of $d = 0.1$.

Preliminary Risk-Cost Trade-off Analysis

Figure 3.5 contains a scatter plot with the cost and safety measures for n evaluations of the LQR, CBF, and CVaR controllers. While the LQR controller is highly optimal in terms of the cost associated with the trajectory along all system states, it offers no regard for safety. The CBF method deviates from the nominal controller in a way that sacrifices some cost optimality, but in return can avoid unsafe states that may be present in the LQR trajectory. While the CBF method does improve the measure of safety, in the presence of noise it may still select control inputs that result in unsafe states. The CVaR barrier function accounts for noise disturbances in a probabilistic approach, resulting in higher safety measures and often reaching near the same level of cost optimality as the CBF method. For the purpose of this research, the CVaR barrier function renders the other methods examined obsolete.

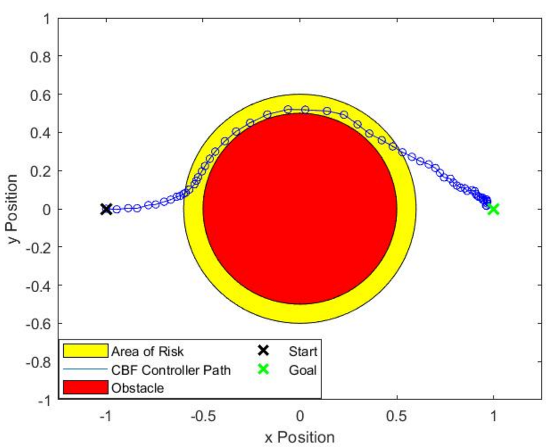


(a) Output Path

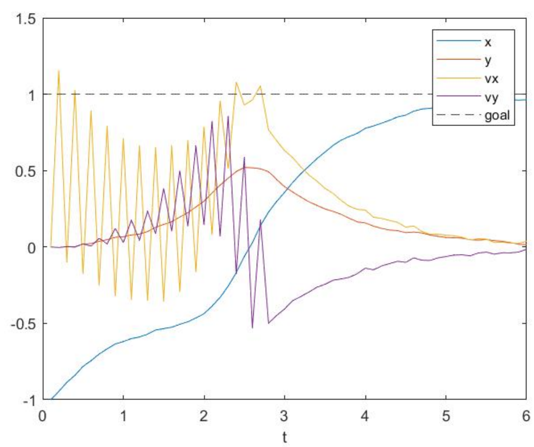


(b) Trajectory

Figure 3.2: Simulation of Nominal Controller

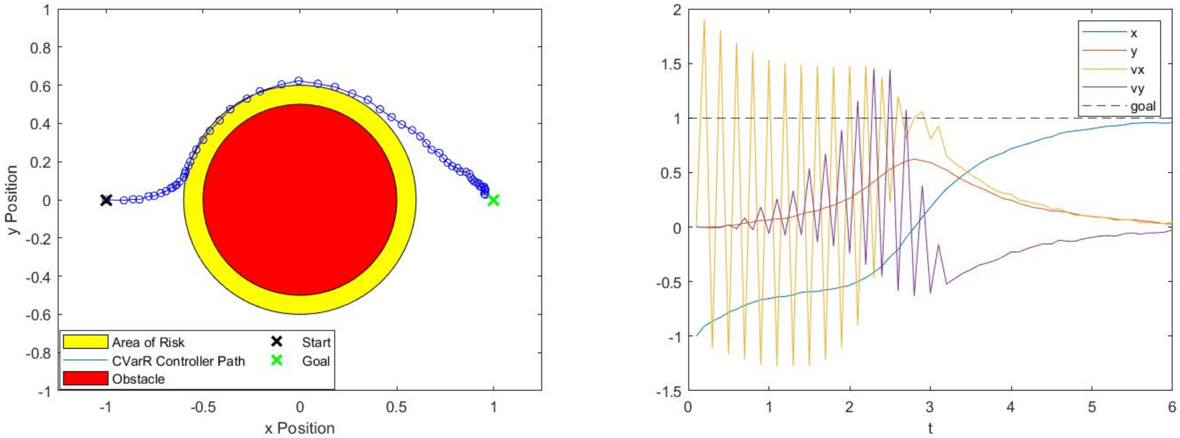


(a) Output Path



(b) Trajectory

Figure 3.3: Simulation of CBF Controller



(a) Output Path

(b) Trajectory

Figure 3.4: Simulation of CVaR Controller

So in the next sections, we only examine the CVaR barrier function in comparison to our method.

3.2.2 Combined Cost Method

Results from work thus far have demonstrated a clear trade-off between cost and risk by our definitions of each. Oftentimes a control method will prioritize cost or risk, so methods are selected on a situational basis. What we propose is a control method that can find an optimal solution for any given preference towards one or the other. This preference is specified by a weighting parameter $\lambda \in [0,1]$, where $\lambda = 0$ corresponds to a cost-averse, risk-neutral controller, and $\lambda = 1$ corresponds to a risk-averse, cost-neutral controller. Values in between provide some trade-off between the two utilities. To provide a trade-off between cost and risk we propose an optimization problem that minimizes a combination of cost and risk weighted by λ. We will refer to this as the "Combined Cost" function. A safety constraint may be characterized as $\mathbb{E}[\text{risk}(K)] \leq \varepsilon$, where $\varepsilon \in [0,1]$ is the given level of acceptable risk. For every ε, there is an associated λ appropriate for use in the Combined Cost function. Using the given upper bound to risk, we can manipulate the definition of risk from a cumulative product, to a cumulative sum of logarithms that still guarantees the initial constraint of $\mathbb{E}[\text{risk}(K)] \leq \varepsilon$. We start by taking the complement of both sides of the inequality:

$$\mathbb{E}[1 - \text{risk}(K)] \geq 1 - \varepsilon.$$

In other words, if risk is less than the bound, we can infer that the probability of success must be greater than the complement of the risk bound. Referring back to Equation 3.1 in

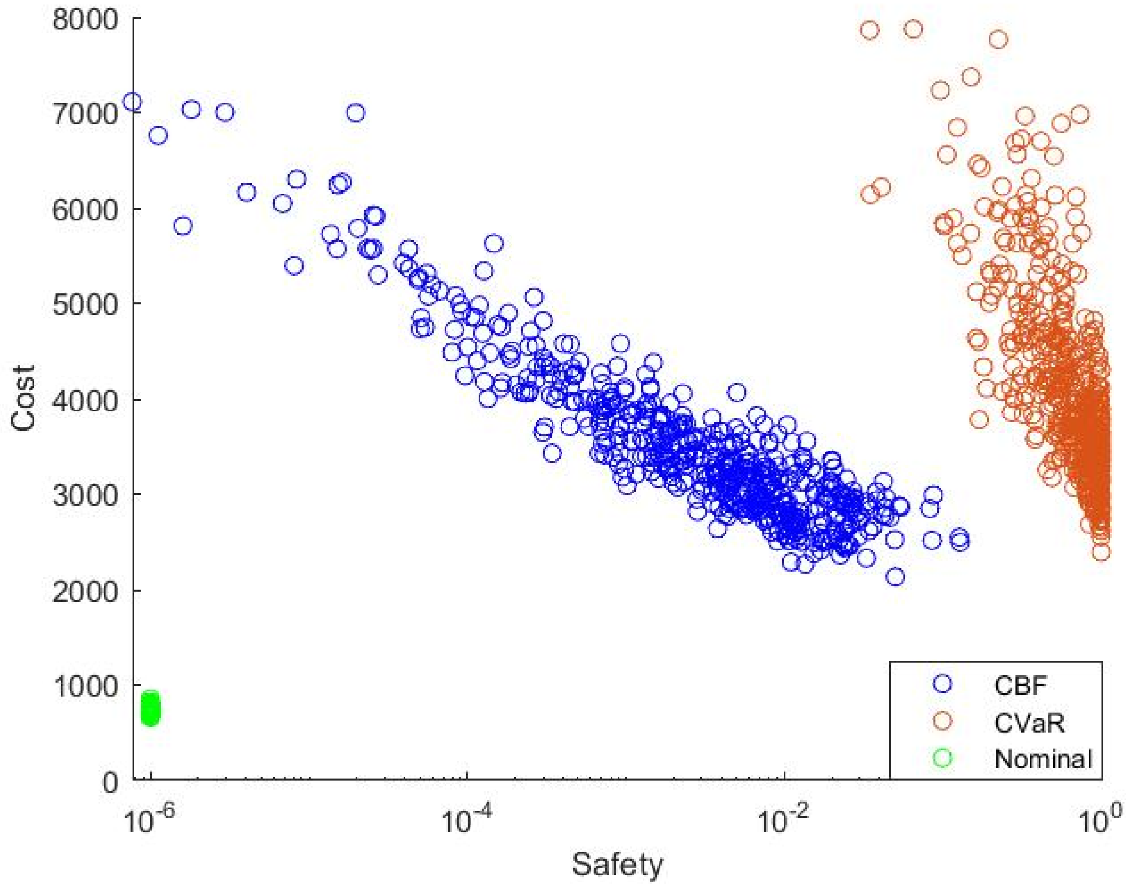


Figure 3.5: A scatter plot where each point represents the cost and safety resulting from the trajectory of a controller

Section 3.1.2. we can rewrite this inequality as:

$$\mathbb{E}\left[\prod_t (1 - r_t(x_t, u_t))\right] \geq 1 - \varepsilon.$$

Finally, by taking the logarithm of both sides we can ensure the logarithm still is monotonically increasing and the following inequality holds true:

$$\log \mathbb{E}\left(\prod_t (1 - r_t(x_t, u_t))\right) \geq \log(1 - \varepsilon).$$

When the stage risk is the exponential of a quadratic

$$r_t(x_t, u_t) = 1 - \exp\left(-\begin{bmatrix} x_t \\ u_t \end{bmatrix}^\top Q_t^{risk} \begin{bmatrix} x \\ u \end{bmatrix} + (R_t^{risk})^\top \begin{bmatrix} x_t \\ u_t \end{bmatrix} + S_t^{risk}\right),$$

Using the product rule of logarithms we can express the left-hand side of the inequality as a cumulative sum rather than a cumulative product. We can then rearrange this inequality to be in terms of the LEQG function outlined in Section 2.5.1. If we denote

$$J_t^{risk}(x_t, u_t) = \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\top Q_t^{risk} \begin{bmatrix} x \\ u \end{bmatrix} + (R_t^{risk})^\top \begin{bmatrix} x_t \\ u_t \end{bmatrix} + S_t^{risk},$$

we can infer

$$\log \mathbb{E} \exp \left(-\sum_t J_t^{risk}(x_t, u_t) \right) \geq \log(1 - \varepsilon),$$

or equivalently

$$\log \mathbb{E} \exp \left(\sum_t J_t^{risk}(x_t, u_t) \right) \geq -\log(1 - \varepsilon).$$

With risk now in this form, we can form a Combined Cost function J as a weighted combination of cost and risk formulated as an LEQG objective.

$$J = (1 - \lambda)\mathbb{E} \left[\sum_t c_t(x_t, u_t) \right] + \lambda \log \mathbb{E} \left[\exp \left(\sum_t J_t^{risk}(x_t, u_t) \right) \right]$$

To minimize this function and approximate a trade-off between cost and risk using the Combined Cost method, the following criteria must hold:

- The system dynamics are modeled linearly.
- The cost function is quadratic.
- The risk function is an exponential of a quadratic.
- The process noise is Gaussian.

Under these assumptions J is guaranteed to be a linear quadratic function. If we define the risk parameters Q, R, and S as a convex combination of cost and risk as:

$$Q_t = (1 - \lambda)Q_t^{cost} + \lambda Q_t^{risk},$$

$$R_t = (1 - \lambda)R_t^{cost} + \lambda R_t^{risk},$$

$$S_t = (1 - \lambda)S_t^{cost} + \lambda S_t^{risk},$$

we can directly apply the LQG method outlined in Section 2.5.1.

Similar to the methods [2, 18, 30], the combined cost method makes use of a weighing parameter to directly balance the trade-off between cost and risk. However [2,18,30] are based in intelligent control, while the Combined Cost method is based in optimal control. When there is no noise, the combined cost method is optimal due to the fact that there would no longer be expectations in J. Without the expectations in J the log and exponential functions cancel each other out and the optimization problem can be solved exactly. In the presence of noise this method is sub-optimal, however we wish to demonstrate the benefits of this method regardless, as it is our understanding that an optimal solution is not yet known for a problem of this nature.

3.2.3 Scenario 2: 1D LQG

Unlike the CBF and CVaR barrier function, the method proposed in this research is much more specific in terms of potential applications. The use of barrier functions in the previously implemented methods allows for multiple sets of risks in clearly defined areas. In the path planning example described in the completed work section, there is a single unsafe set. Risk is centralized around the unsafe set, and risk decreases as the states move away from the unsafe set. However, in an LQG scenario the risk function decreases in relation to a single equilibrium point. An example showing how the first scenario cannot meet the LQG criteria is shown in Figure 3.6.

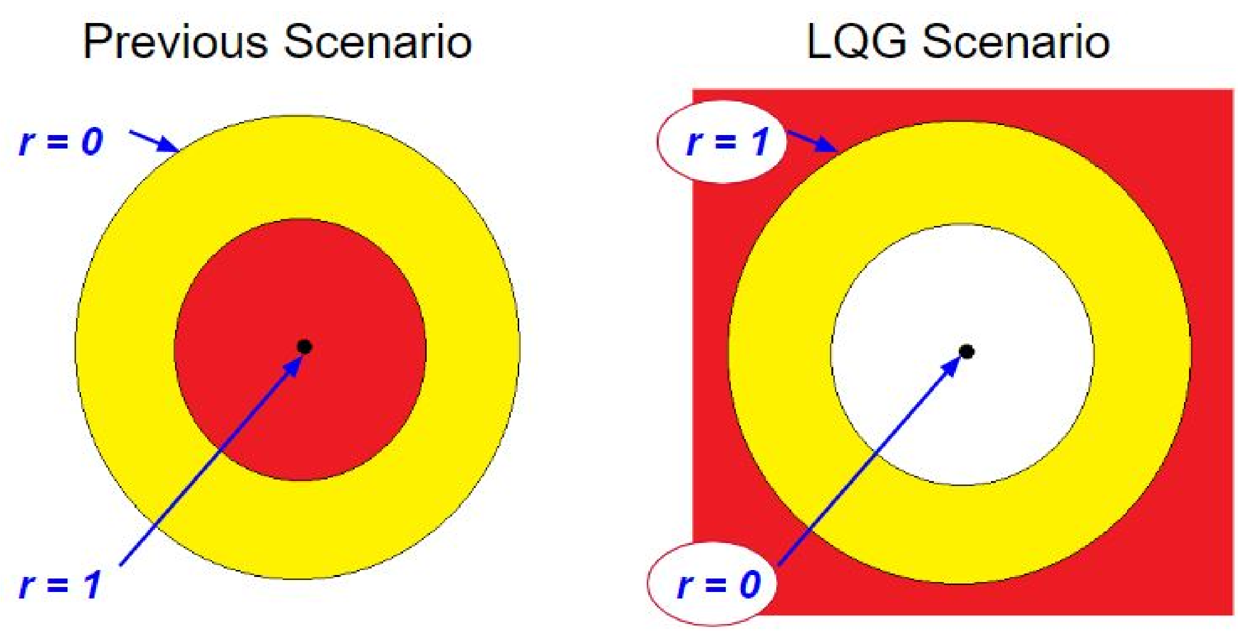


Figure 3.6: The pattern risk may follow in the previous scenario vs the LQG scenario

Therefore, the second scenario must differ from the classic obstacle avoidance example. We will instead look at a situation where the goal is for an agent to reach an equilibrium point without travelling too fast.

System Dynamics

We could use the same double integrator system as defined in Scenario 1, but for the sake of simplicity we will be only be considering one direction of movement for Scenario 2. After reducing the dimension of the system dynamics in Scenario 1 we arrive at the following state space representation:

$$F(x_t, u_t, w_t) = \begin{bmatrix} 1 & t_s \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} \frac{1}{2}t_s^2 \\ t_s \end{bmatrix} u_t + w_t$$

with states p_x and v_x. This system has an equilibrium point at the origin and an initial state $\left[\begin{smallmatrix} px_0 \\ vx_0 \end{smallmatrix}\right]$.

Cost

The goal in this scenario is to find the lowest cost trajectory from the initial state to the equilibrium point while adhering to some risk limit, where cost is purely dependent on position and risk is purely dependent on velocity. Recall that in the previous scenario, cost was a function of all states and risk was dependent on the distance from a barrier. Like in the previous scenario, cost will still be in the linear quadratic form. However now the LQG weighing matrices will heavily prioritize position. The state matrix used in the implementations of Scenario 2 is

$$Q_t^{cost} = \begin{bmatrix} 1 & 0 \\ 0 & 10^{-5} \end{bmatrix}$$

where 10^{-5} is in the same scale of magnitude as the simulated noise.

Risk

While the cost function can remain in the same form as in Scenario 1, for Scenario 2, the risk function must be adapted in order to fit the LQG criteria. In Section 3.2.2 we defined the cost associated with risk in the Combined Cost function as $\log(1 - r_t(x_t, u_t))$. For this to evaluate to the linear quadratic format needed for the Combined Cost function, the stage risk must take the following form:

$$r_t(x_t, u_t) = 1 - e^{-\begin{bmatrix} x_t \\ u_t \end{bmatrix}^\top Q_t^{risk} \begin{bmatrix} x \\ u \end{bmatrix} + (R_t^{risk})^\top \begin{bmatrix} x_t \\ u_t \end{bmatrix} + S_t^{risk}}$$

As mentioned previously, the risk in Scenario 2 is dependent upon velocity. Higher magnitude velocity corresponds to more risk. The following state matrix places the weight for cost associated with risk upon velocity:

$$Q_t^{risk} = \begin{bmatrix} 10^{-5} & 0 \\ 0 & 1 \end{bmatrix}.$$

Implementation

For the simulation in Scenario 2, we looked at a time horizon of 100 steps with step size $t_s = 0.1$. We simulated Gaussian process noise with covariances 3×10^{-5} and 5×10^{-5} in states p_x and v_x respectively. The initial conditions were $\left[\begin{smallmatrix} px_0 \\ vx_0 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right]$. As mentioned previously, we defined the LQR state cost and risk matrices Q_t^{cost} and Q_t^{risk} to associate cost with position and risk with velocity. We left the cross-correlation matrices null as in the previous scenario. Since the cost and risk definitions in this scenario do not directly depend upon the input, but rather the system states, we also initialized R_t^{cost} and R_t^{risk} as null for simplicity.

To examine the performance of the controllers it is necessary to calculate the cost for varying levels of risk. In the Combined Cost method, we varied the weighting parameter, λ and found the solution via LQG for various levels of risk. In the CVaR controller has multiple parameters that can influence the strictness of the safety constraint, primarily the stringency α, the confidence level β, and the barrier function itself. Since we are defining stage risk in terms of the magnitude of velocity, it makes sense to select a barrier function that imposes a velocity limit as a safety constraint. We can then hold α and β constant and vary the velocity limit, which we will call V_{lim}, in order to establish a spectrum of risk for our analysis. The barrier function used for the implementation of the CVaR controller in Scenario 2 is $h(x_t) = V_{lim} - |v_{xt}|$. For simplicity, we selected $\alpha = 0$ for the stringency level. This means that all positive evaluations of the barrier function satisfy the safety constraint. We selected a confidence level $\beta = 0.5$ for the initial simulations, signifying a risk-aware setting of the CVaR controller. We then solved the CVaR barrier function optimization problem for various velocity limits corresponding to risk levels in between 0 and 1 to collect the necessary data for our results.

We generated results for these simulations in two different ways. In the first analysis we wanted to look at a direct comparison between the performance of the controllers at specific risk levels. To do this, we first had to find values of both λ in the Combined Cost method and V_{lim} in the CVaR barrier function that correspond to a given risk limit of the total risk ε. For the Combined Cost method, we could relate λ to ε as follows. Recall this step from the derivation of the combined cost function:

$$\log \mathbb{E}\left(\prod_t (1 - r_t(x_t, u_t))\right) \geq \log(1 - \varepsilon).$$

We can equivalently express the expected value of cumulative probability in terms of the average stage risk across the time horizon

$$T_L \log(1 - r_{avg}) \geq \log(1 - \varepsilon),$$

where T_L is the length of the trajectory, and r_{avg} is the average stage risk. In other words, if we can find r_{avg} so that if the stage risk was r_{avg} along the entire trajectory the total risk would still be below ε, then the constraint is guaranteed to hold. Solving for the average stage risk, we get

$$r_{avg} \leq 1 - \exp\left(\frac{1}{T_L} \log(1 - \varepsilon)\right).$$

We then conduct a bisection search to find the value of λ that returns an average stage risk that satisfies this constraint.

After finding the appropriate value of λ that when used in control produces a trajectory with total risk ε, we must find a value of V_{lim} that produces a trajectory with approximately the same amount of risk. This is similarly achieved by using a bisection search of the CVaR barrier function to return the appropriate V_{lim}. Figure 3.7 contains the results from one iteration of analysis using the bisection search to find a direct comparison between the cost of both controllers at a range of risk levels in between 0 and 1.

This analysis provides a useful visualization, but the bisection search requires many iterations of both control algorithms to find the solution at the specified risk level. This is computationally complex, especially for the CVaR method which solves an optimization problem at every point of a given trajectory. For the purpose of providing greater overarching data, in the results in Figure 3.8, we simply generated many data points for each controller varying λ and V_{lim} to produce a scatter plot of data. We then found the expected value of each data set amongst the varying incidents of noise to make our comparison. Figure 3.8 contains data from 980 trajectories. The shaded regions represent the area in which the majority of the data lies for each controller. When examining the expected value for each data-set it is evident that for every level of risk the Combined Cost controller is expected to produce a lower cost than the CVaR barrier function controller.

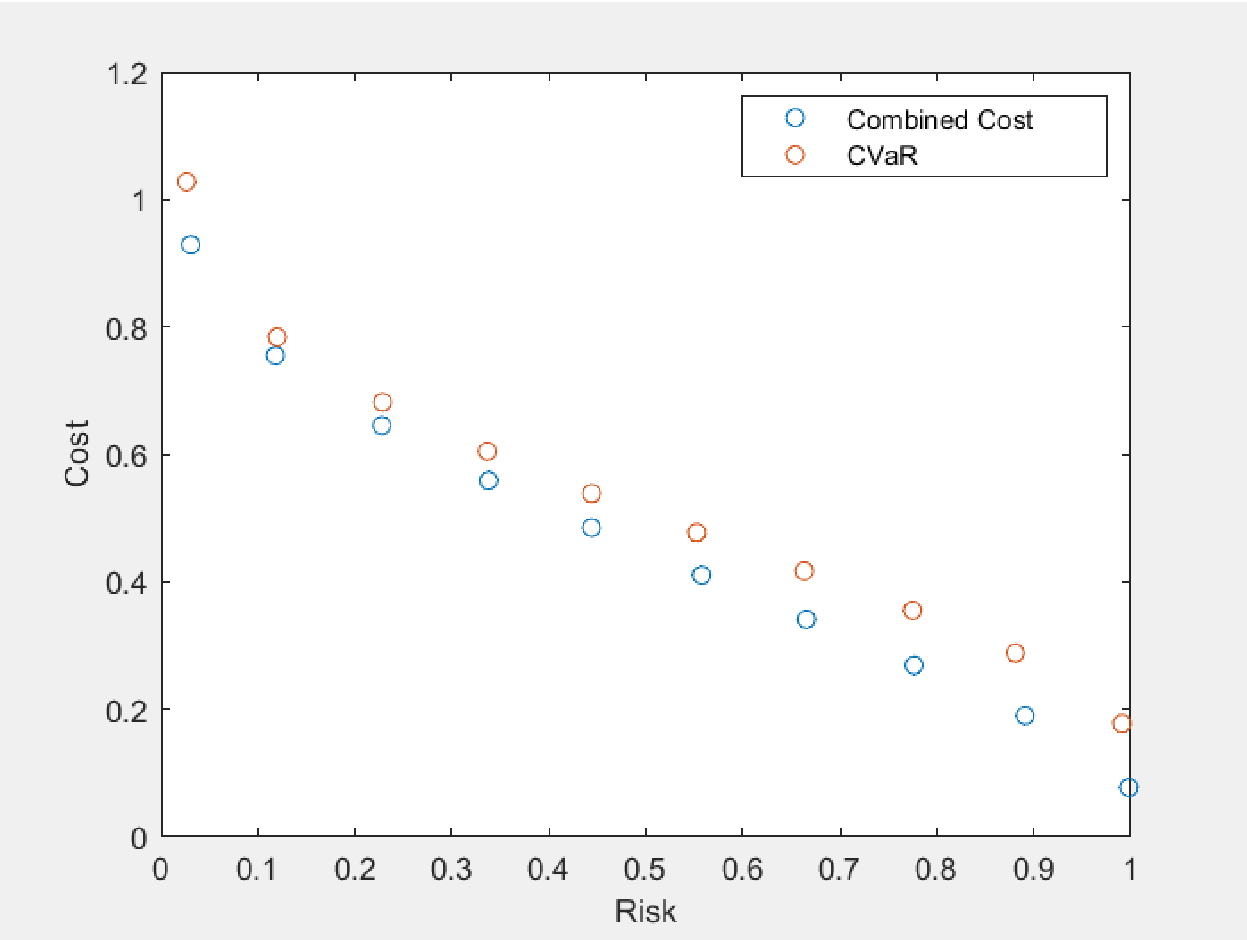


Figure 3.7: One iteration of the bisection search to compare the Combined Cost and CVaR controllers with $\beta = 0.5$

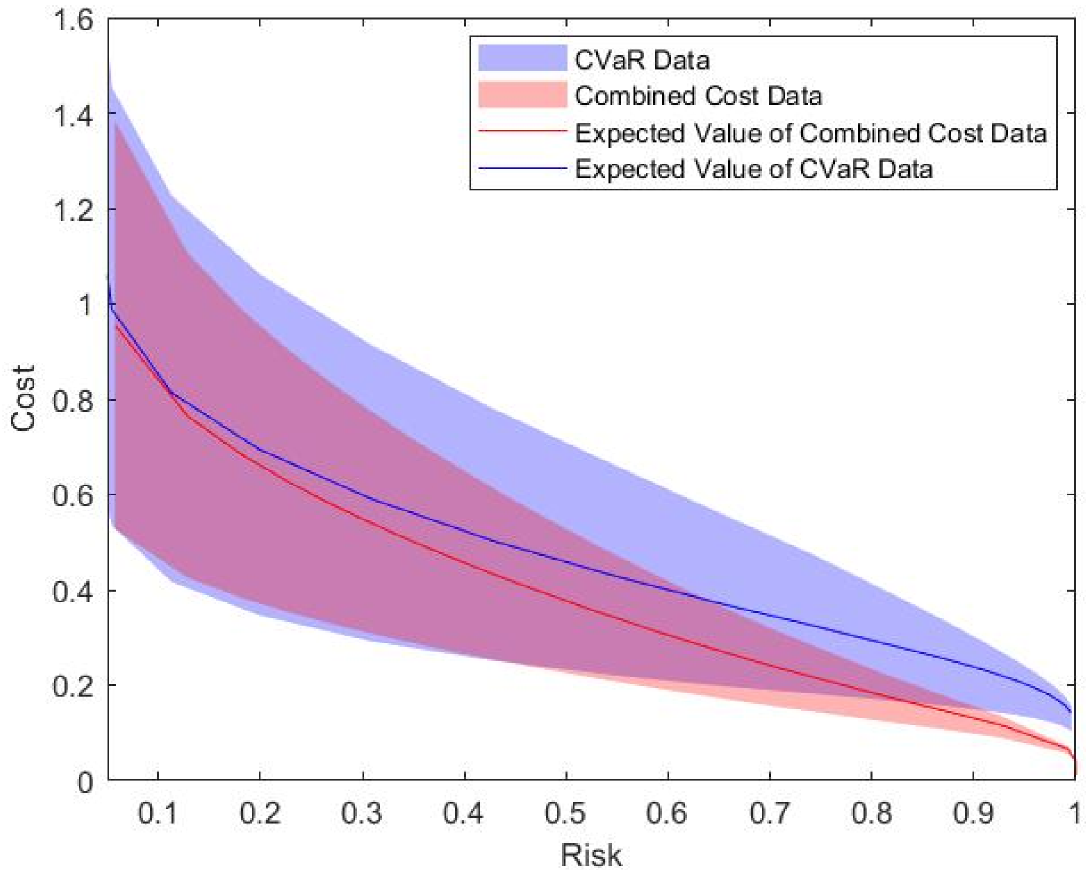


Figure 3.8: Sampled Data of Combined Cost and CVaR controllers with $\beta = 0.5$

4

Conclusion

In the real world, there is always some element of stochasticity to a system, thus it is important to design robust controllers. From optimal control based on explicit mathematical solutions, to intelligent control methods that rely on data and pattern recognition to make decisions, there are many ways to design controllers with both cost and safety in mind. The majority of these methods offer some tunability to allow for various stringency of risk constraints, however our proposed method aims to provide a direct mediation between the two goals with a weighting parameter. There are existing methods that make use of a weighting parameter, but the Combined Cost method offers a novel approach of doing this in the LQG setting.

This method is only exactly optimal in the deterministic case. However, when compared to other optimal control methods, such as the nominal LQR controller, the CBF, and the CVaR barrier function, the Combined Cost controller has proven to produce trajectories with a lower expected value of cost for every given level of risk in the presence of noise. While the CVaR barrier function produced similar expected costs for levels of risk around 20% and below, it is far more computationally complex than the other methods. The Combined Cost controller provides the greatest results at nearly the computational simplicity of the nominal LQR controller, which does not even account for safety. While in scenarios meeting the LQG criteria the Combined Cost method yields exemplary results. However, this comes with limited versatility. The other methods listed can be applied in many scenarios, while the Combined Cost method can only be applied under very specific circumstances.

The key to improving this method lies in exploring ways in which we can improve its optimality under the condition of noise, as well as expand its versatility. While we examined the application of trading off cost and safety in this work, in reality this method could be used to trade off any two utilities given they are a quadratic function or the exponential of a quadratic function. Additionally, by looking at ways we can adapt or model a given problem scenario to fit the LQG criteria we can increase the applicability of the Combined Cost method. While the Combined Cost controller has displayed advantages within the context of this work, there is still room for further analysis and growth in the future.

References

[1] Lavon B. Page and Jo Ellen Perry. Reliability, recursion, and risk. volume 98, pages 937–946. Taylor & Francis, 1991.

[2] Mohamadreza Ahmadi, Xiaobin Xiong, and Aaron D. Ames. Risk-averse control via cVaR barrier functions: Application to bipedal robot locomotion. volume 6, pages 878–883, 2022.

[3] International Electrotechnical Commission. Degrees of protection provided by enclosures (ip code), 2013. Edition 2.2.

[4] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications, 2019.

[5] Andrew Clark. Control barrier functions for stochastic systems, 2020.

[6] Merve Öztekin, Oleh Kiselychnyk, and Jihong Wang. Nonlinear optimal control for interior permanent magnet synchronous motor drives. In *2022 European Control Conference (ECC)*, pages 590–595, 2022.

[7] Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Risk-Sensitive motion planning using entropic value-at-Risk. In *2021 European Control Conference (ECC)*, pages 1726–1732, 2021.

[8] Margaret P. Chapman and Laurent Lessard. Toward a scalable upper bound for a CVaR-LQ problem. volume 6, pages 920–925, 2022.

[9] Pouria Tooranjipour, Bahare Kiumarsi, and Hamidreza Modares. Risk-aware safe optimal control of uncertain linear systems. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1–5, 2022.

[10] Sleiman Safaoui, Lars Lindemann, Iman Shames, and Tyler H. Summers. Risk-bounded temporal logic control of continuous-Time stochastic systems. In *2022 American Control Conference (ACC)*, pages 1555–1562, 2022.

[11] Andrew Singletary, Mohamadreza Ahmadi, and Aaron D. Ames. Safe control for non-linear systems with stochastic uncertainty via risk control barrier functions. volume 7, pages 349–354, 2023.

[12] Mohamadreza Ahmadi, Andrew Singletary, Joel W. Burdick, and Aaron D. Ames. Safe policy synthesis in multi-agent POMDPs via discrete-time barrier functions. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4797–4803, 2019.

[13] Masahiro Ono and Brian Charles Williams. An efficient motion planning algorithm for stochastic dynamic systems with constraints on probability of failure. In *AAAI*, 2008.

[14] Xiaobin Xiong and Aaron Ames. 3-d underactuated bipedal walking via H-LIP based gait synthesis and stepping stabilization, 2022.

[15] Hucheng Jiang, Tianjiao An, Bing Ma, Yuanchun Li, and Bo Dong. Value iteration-based decentralized fuzzy optimal control of modular reconfigurable robots via adaptive dynamic programming. In *2022 5th International Conference on Robotics, Control and Automation Engineering (RCAE)*, pages 186–190, 2022.

[16] Kehan Long, Vikas Dhiman, Melvin Leok, Jorge Cortés, and Nikolay Atanasov. Safe control synthesis with uncertain dynamics and constraints. volume 7, pages 7295–7302, 2022.

[17] Jason J. Choi, Fernando Castañeda, Claire J. Tomlin, and Koushil Sreenath. Reinforcement learning for safety-Critical control under model uncertainty, using control lyapunov functions and control barrier functions. volume abs/2004.07584, 2020.

[18] C. Pace and D.W. Seward. A model for autonomous safety management in a mobile robot. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, volume 1, pages 1128–1133, 2005.

[19] Aditya Undurti and Jonathan P. How. A decentralized approach to multi-agent planning in the presence of constraints and uncertainty. In *2011 IEEE International Conference on Robotics and Automation*, pages 2534–2539, 2011.

[20] Pedro Rodrigues Quemel e Assis Santana, Sylvie Thiébaux, and Brian Williams. RAO*: An algorithm for chance-constrained POMDP's. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016.

[21] Mohamadreza Ahmadi, Nils Jansen, Bo Wu, and Ufuk Topcu. Control theory meets POMDPs: A hybrid systems approach. *IEEE Transactions on Automatic Control*, 66(11):5191–5204, 2021.

[22] Azza Allouch, Anis Koubâa, Mohamed Khalgui, and Tarek Abbes. Qualitative and quantitative risk analysis and safety assessment of unmanned aerial vehicles missions over the internet. *IEEE Access*, 7:53392–53410, 2019.

[23] Legal information institute - VaR-based measure. Accessed: April 17, 2023.

[24] Chuanning Wei, Michael Fauß, and Margaret P. Chapman. CVaR-based safety analysis in the infinite time horizon setting. In *2022 American Control Conference (ACC)*, pages 2863–2870, 2022.

[25] Mohamadreza Ahmadi, Anushri Dixit, Joel W. Burdick, and Aaron D. Ames. Risk-averse stochastic shortest path planning. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 5199–5204, 2021.

[26] Samantha Samuelson and Insoon Yang. Safety-aware optimal control of stochastic systems using conditional value-at-risk. In *2018 Annual American Control Conference (ACC)*, pages 6285–6290, 2018.

[27] Russ Tedrake. *Underactuated Robotics*. 2023.

[28] Peter Whittle. *Optimal Control: Basics and Beyond*, pages 295–320. John Wiley Sons, Inc., 1996.

[29] Anastasios Tsiamis, Dionysios S. Kalogerias, Luiz F. O. Chamon, Alejandro Ribeiro, and George J. Pappas. Risk-Constrained linear-Quadratic regulators. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3040–3047, 2020.

[30] Zonghua Zhang, Xiaodong Lin, and Pin-Han Ho. Measuring intrusion impacts for rational response: A state-based approach. In *2007 Second International Conference on Communications and Networking in China*, pages 317–321, 2007.

[31] Taisuke Kobayashi, Takumi Aotani, Julio Rogelio Guadarrama-Olvera, Emmanuel Dean-Leon, and Gordon Cheng. Reward-punishment actor-critic algorithm applying to robotic non-grasping manipulation. In *2019 Joint IEEE 9th International Conference*

on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), pages 37–42, 2019.