Control, Estimation, Modeling

# Compiled Notes on Control Theory

June 2025

## Table of Contents

# 1 Goals of Control Theory

Control theory is a branch of engineering that deals with the behavior of dynamical systems and how to influence that behavior through inputs. Typical considerations are:

**Stability** Ensuring the system remains bounded and predictable over time, and preventing oscillations or otherwise chaotic behavior.

**Controllability** Determining whether a system can be driven from an initial state to a desired final state using the available control inputs.

**Observability** Assessing how much of the internal state of a system can be inferred from its outputs. Crucial for systems where direct measurement of all variables isn't possible.

**Optimality** Achieving the desired behavior with minimal cost (energy, time). Involves balancing between speed, accuracy, and resource usage.

**Robustness** Maintaining performance despite uncertainties, disturbances, or modelling inaccuracies. The controller must be able to handle real world imperfections!

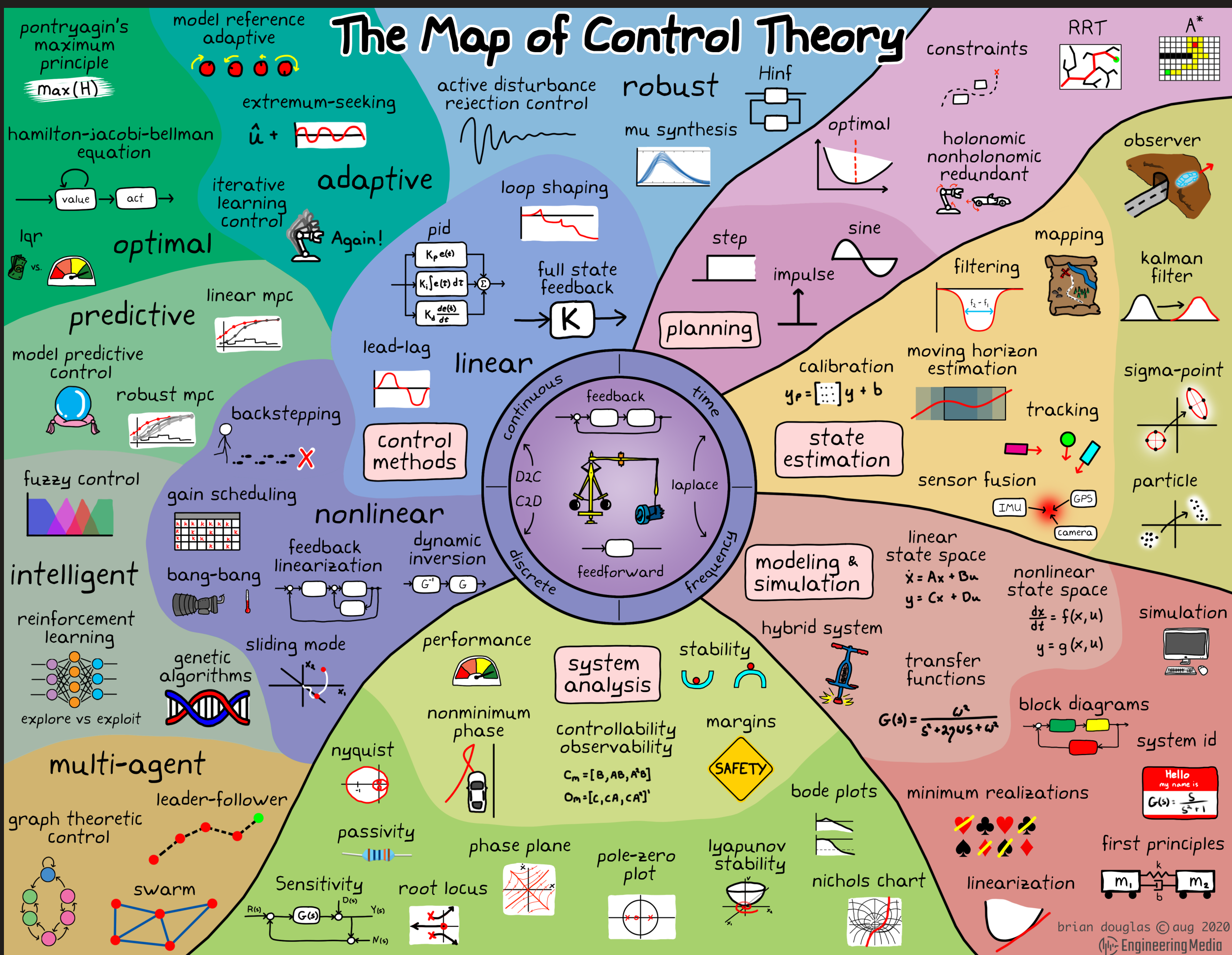


Figure 1: Brian Douglas' Concept Map of Control Theory

Control theory is a interdisciplinary topic. It can be applied to mechanical, electrical, thermal, fluid, really any system.

## 1.1 Applications to Aerospace

The main application of control theory in aerospace is in Guidance, Navigation, and Control (GNC). In order from most broad to least:

**Navigation (Statistics)** ”Given the measurements we have, where do we think we are?”
**Guidance (Optimization)** ”Given where we want to go and where we think we are, which path should we follow?”
**Control (Differential Equations)** ”What effort should we apply and how should we control actuators given where we are and the chosen path?”

All require knowledge of dynamics and linear algebra, and can be further broken down into numerous subcategories.
Control theory offers many benefits in the aerospace application.

**Automation** Remove the need for human intervention, preserving the attention of the operator for more important matters.
**Performance** Operate more effectively than a human operator.
**Safeguards / Protections** Prevent the craft from exceeding design limits.
**Deferred Decision-making** Let the human operator make certain flight critical decisions.

# 2 Background

Some background information must be understood before delving into control theory.

## 2.1 Calculus

I am not trying to rewrite a textbook here, nor do I have to knowledge to. This will just be a quick refresher on the most important topics.

### 2.1.1 *Differentiation*

### 2.1.2 *Integration*

## 2.2 Differential Equations

### 2.2.1 *Ordinary (ODE)*

### 2.2.2 *Partial (PDE)*

### 2.2.3 *Jacobian Matrix*

### 2.2.4 *Convolution*

Convolution integrals are useful for solving initial value problems with general forcing functions.

$$(f * g)(t) = \int_0^t f(t-\tau)g(\tau)d\tau \tag{2.1}$$

### 2.2.5 *Fourier Transforms*

Any periodic function can be represented by a Fourier series (series of periodic functions).

For periodic functions that decay to zero, we can use the Fourier transform.

$$f(w) = \int_{-\infty}^{\infty} f(t)e^{-iwt}dt \tag{2.2}$$

This transform converts a continuous time domain signal into a continuous frequency domain signal. In other words, it converts a signal into its component frequencies.

### 2.2.6 *Laplace Transforms*

The Laplace transform is a more general case of the Fourier transform.

**Transfer Functions** Applying the Laplace transform to a time domain function will produce the frequency domain representation of that function, also known as a transfer function.

**Geometry of Transfer Functions**

### 2.2.7 *Delayed Differential Equations*

### 2.2.8 *Poles & Zeros*

The poles and zeros of a transfer function provide some critical information

## 2.3 Linear Algebra

### 2.3.1 *Basic Operations*

### 2.3.2 *Matrix Inversion*

### 2.3.3 *Eigenvectors & Eigenvalues*

## 2.4 Mechanics (Newtonian)

Understanding the physics that inform the behavior of a system is critical to modeling and controlling it.

**Newton's 1st Law**
**Newton's 2nd Law**
**Newton's 3rd Law**

### 2.4.1 *Linear*

### 2.4.2 *Rotational*

**Coriolis Force** The presense of rotation adds an additional (faux) force. This is described further in (7.1.5).

### 2.4.3 *Electrical*

Electrical circuits can be modeled very similarly to the mechanical system. The classic example is RLC (resistor-inductor-capacitor) circuits.

## 2.5 Lagrangian Mechanics

Lagrangian mechanics are preferable to use for dynamical systems that have multiple components whose interaction is complicated to express (eg. multi-body systems). This is especially true if internal reactions are not of immediate interest.

$$\text{Action: } S = \int_{t_1}^{t_2} L(q, \dot{q}) dt \tag{2.3}$$

Where $L(q, \dot{q})$ is the Lagrangian and $q \in \mathbb{R}^{nq}$ is a vector of generalized degrees of freedom (positions, angles). $\dot{q}$ is the vector of corresponding rates of change of the degrees of freedom (linear + angular velocities).

$$L(q, \dot{q}) = T(q, \dot{q}) - V(q) \tag{2.4}$$

$T(q, \dot{q})$ is the kinetic energy function, $V(q)$ is the potential energy function. Thus the Lagrangian is the difference between the system's total kinetic and total potential energy.

**Principle of Least Action** A conservative system (one that does not gain or lose energy) starting in state $q(t_1)$ at time $t_1$ and ending at state $q(t_2)$ at time $t_2$ follows a path that makes the action $S$ stationary (first derivative goes to zero). This is the path of least action. While the path of least action minimizes the action the vast majority of the time, the stationary condition does not guarantee minimization.

**Euler-Lagrange Equations for Conservative Systems** The path of least action can be thought of as a continuous steam of states $q(t)$ over the time interval. It must satisfy the Euler-Lagrange Equations.

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_j}) - \frac{\partial L}{\partial q_j} = 0, \quad j \in \{1, ..., n_q\} \tag{2.5}$$

This yields $n_q$ equations of motion, one corresponding to each of the generalized degrees of freedom. This we obtain all the equations of motion necessary to describe a conservative (energy-preserving) dynamical system.

**Euler-Lagrange Equations for Non-conservative Systems** We can modify Eq. 2.5 to account for energy dissipation (eg. friction) and energy (eg. forcing).

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_j}) - \frac{\partial L}{\partial q_j} = Q_j - \frac{\partial R}{\partial \dot{q}_j} \tag{2.6}$$

where $Q_j := Q_j(t)$ is the forcing function and $R := R(\dot{q})$ is the energy dissipation function. If we plug in the definition for $V(q)$ from the Least Action Principle, we arrive at:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_j}) + \frac{\partial R}{\partial \dot{q}_j} + (\frac{\partial V}{\partial q_j} - \frac{\partial T}{\partial q_j}) = Q_j \tag{2.7}$$

This is the equation of motion for each degree of freedom.

**Example** We can use the classic example of the mass-spring-damper system. The advantages of the Lagrangian method become clear when an inverted pendulum is added to the system.

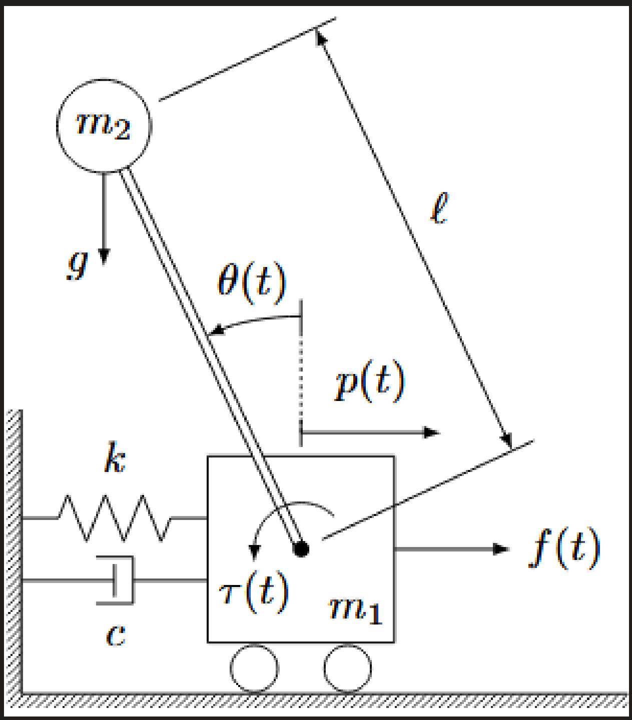


Figure 2: Inverted Pendulum on Cart System

With two bodies (masses) in the system, we can describe the degrees of freedom and the velocities of each as:

$$
\begin{aligned}
q_1(t) &:= p(t), \quad \dot{q}_1(t) = \dot{p}(t), \\
q_2(t) &:= \theta(t), \quad \dot{q}_2(t) = \dot{\theta}(t), \\
Q_1(t) &:= f(t), \quad Q_2(t) := \tau(t), \\
v_1 &= \begin{bmatrix} \dot{p} \\ 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} \dot{p} - \ell\dot{\theta}\cos(\theta) \\ -\ell\dot{\theta}\sin(\theta) \end{bmatrix}
\end{aligned}
$$

Now, the components of the Euler-Lagrange equation (Eq. 2.7) can be found to assemble the complete equations of motion of the system.

$$T_i = \frac{1}{2} m_i v_i^T v_i$$

$$T_1 = \frac{1}{2} m_1 (\dot{p}^2 + 0^2), \quad T_2 = \frac{1}{2} m_2 [(\dot{p} - \ell\dot{\theta}\cos(\theta))^2 + (-\ell\dot{\theta}\sin(\theta))^2]$$

The sum of the kinetic energy of the system is the sum of the kinetic energy of both masses:

$$
\begin{aligned}
T &= T_1 + T_2 \\
&= \frac{1}{2} m_1 \dot{p}^2 + \frac{1}{2} m_2 [\dot{p}^2 - 2\ell\cos(\theta)\dot{p}\dot{\theta} + \ell^2\dot{\theta}^2]
\end{aligned}
$$

We can then differentiate the above equation with respect to the generalized velocities $\dot{q}_i$.

This gives us the following partial derivatives:

$$\frac{\partial T}{\partial \dot{q}_1} = (m_1 + m_2)\dot{p} - m_2 \ell \dot{\theta} cos(\theta) \quad \text{and} \quad \frac{\partial T}{\partial \dot{q}_2} = m_2 \ell [\ell \dot{\theta} - \cos(\theta)\dot{p}]$$

Differentiating again, with respect to time:

$$\frac{d}{dt}(\frac{\partial T}{\partial \dot{q}_1}) = (m_1{+}m_2)\ddot{p}{-}(m_2\ell)(\ddot{\theta}\cos(\theta){+}\sin(\theta)\dot{\theta}^2) \quad \text{and} \quad \frac{d}{dt}(\frac{\partial T}{\partial \dot{q}_2}) = m_2\ell(\ell\ddot{\theta}{-}\ddot{p}\cos(\theta){-}\dot{p}\sin(\theta))$$

Then, we also differentiate the total kinetic enery equation with respect to position:

$$\frac{\partial T}{\partial q_1} = 0 \quad \text{and} \quad \frac{\partial T}{\partial q_2} = m_2 \ell \sin(\theta)\dot{\theta}\dot{p}$$

There are two sinks/sources of potential in the system. One is the spring, and the other is the gravitational potential of $m_2$. Thus, the total potential energy of the system is:

$$V = \frac{1}{2}kq_1 + m_2 g \ell \cos(q_2)$$

This equation is differentiated with respect to position:

$$\frac{\partial V}{\partial q_1} = kp \quad \text{and} \quad \frac{\partial V}{\partial q_2} = -m_2 g \ell \sin(\theta)$$

There is one source of energy dissipation in the system: the damper. The energy dissipated by this component is given by:

$$R = \frac{1}{2}c\dot{p}^2$$

Differentiating this with respect to velocity:

$$\frac{\partial R}{\partial \dot{q}_1} = c\dot{p} \quad \text{and} \quad \frac{\partial R}{\partial \dot{q}_2} = 0$$

Now that all of the derivatives have been solved for, the nonlinear ODEs can be plugged into the Euler-Lagrange Equation, resulting in the coupled equations:

$$\begin{aligned} f &= (m_1 + m_2)\ddot{p} - m_2\ell\cos(\theta)\ddot{\theta} + m_2\ell\sin(\theta)\dot{\theta}^2 + c\dot{p} + kp \\ \tau &= m_2\ell^2\ddot{\theta} - m_2\ell\cos(\theta)\ddot{p} - m_2 g\ell\sin(\theta) \end{aligned}$$

The coupled differential equations can be rearranged to isolate $\ddot{p}$ and $\ddot{\theta}$. In order to do this, the constants $\alpha(\theta)$ and $\beta(\theta)$ are defined as follows:

$$\alpha(\theta) := \frac{\cos(\theta)}{\ell} \quad \text{and} \quad \beta(\theta) := \frac{m_2\ell\cos(\theta)}{m_1 + m_2}$$

These constants are used to eliminate the second derivative term in the coupled differential equations. Through some algebraic manipulation, it can be shown that the equations with the isolated second derivative are given by:

$$\ddot{p} = \frac{m_2 \ell \sin(\theta)[\alpha(\theta) g - \dot{\theta}^2] - c\dot{p} - kp + f + \alpha(\theta)\tau}{m_1 + m_2[1 - \alpha(\theta)\ell \cos(\theta)]}$$

$$\ddot{\theta} = \frac{m_2 \ell \sin(\theta)[g - \beta(\theta)\dot{\theta}^2] + \beta(\theta)[f - c\dot{p} - kp] + \tau}{m_2 \ell[\ell - \beta(\theta)\cos(\theta)]}$$

These are the final equations of motion, without solving for the reaction force of the pendulum on the cart!

# 3 System Identification & Analysis

Before we can understand how to control a system, we must first understand how that system changes in response to input. System identification is the process by which a mathematical model of the dynamical system can be constructed. Designing a controller is infinitely easier with such a model.

> The system (excluding the controller) is referred to as the ”plant”.

## 3.1 Linear Regression

## 3.2 Linear, Time-Invariant Systems (LTI)

## 3.3 Linear, Time-Varying Systems (LTV)

## 3.4 State-space Models

## 3.5 Modal Analysis

Eigenvalues (2.3.3) come in handy for understanding how a system reacts to input.

### 3.5.1 *Bounded Input, Bounded Output Stability*

## 3.6 Frequency Domain Methods

# 4 Linear Control Methods

## 4.1 Open Loop

The system identification reveals how the inputs affect the outputs and the state of the system. We can work backwards from the desired output to find the required input.

**Example** In the context of a car's cruise control, we would like to achieve a desired speed with our throttle input. With a model of the car, we can determine the effect of the factors at play in the system: Throttle response, engine force, drag force, tire friction, etc. Treating the system as an equation and solving through, the throttle input required can be solved.

## 4.2 Closed Loop (Feedback)

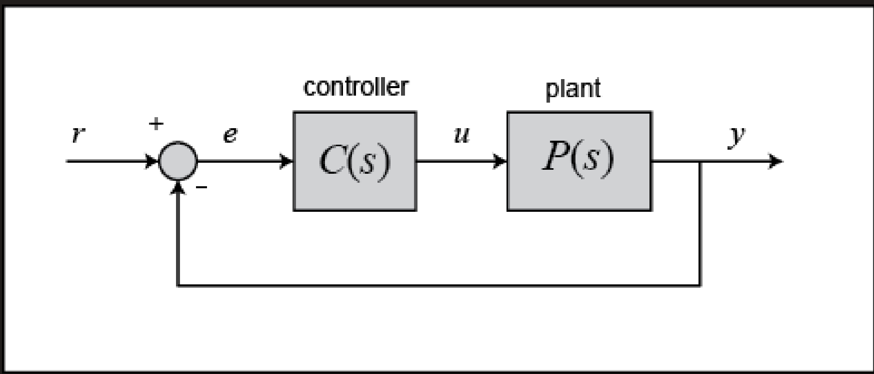


Figure 3: Block Diagram of Feedback Loop

**Example** Returning to the example of the cruise control, imagine that the same system (plant + controller) encounters a hill. Climbing up the slope changes the effect of the gravity force, meaning that our previous throttle input is not longer bringing the desired results. If a feedback system is implemented, the error between the desired speed and the actual speed can be calculated. With this information available, we can design our controller to account for disturbances, such as that hill.

### 4.2.1 *Purpose*

Note: Feedback control systems may be myopic! In many designs, they do not have the information or sophistication to understand that current decisions might make control of the vehicle impossible later on!

## 4.3 Bode Plots

## 4.4 PID Control

The PID controller is one of the simplest types of controllers.

**Proportional Gain (P term)** Applies a control input proportional to the amount of error.

**Integral Gain (I term)** Applies a control input proportional to the integral of error (sum of error over time). This term is useful for countering steady-state error.

**Derivative Gain (D term)** Applies a control input proportional to the rate of change of the error. This term is useful for countering overshoot.

The PID controller can also be represented as a transfer function.

$$K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s} \tag{4.1}$$

### 4.4.1 *Integrator Anti-windup Term*

In certain cases where the error does not correct rapidly, the I term may ”wind up” (error over time builds up, resulting in large corrections). The large correction applied may destabilize the system. To counter this, an ”anti-windup” term can be used, to limit the sum of error the integral term sees.

**Example**

## 4.5 Higher-order Controllers

### 4.5.1 *Lead-Lag Control*

## 4.6 Feedforward

Feedforward control can address some of the drawbacks of feedback control, such as the case of myopia (4.2.1). For this reason, feedforward is sometimes referred to as guidance.

## 4.7 Designing the Controller

### 4.7.1 *Frequency Response Method*

### 4.7.2 *Loop-Shaping Method*

Loop-shaping is somewhat of an enhancement to the frequency response method, using the Nyquist plot in addition to the Bode plot.

### 4.7.3 *Root Locus Method*

## 4.8 Multi-Input, Multi-Output (MIMO)

# 5 Nonlinear Methods

A nonlinear plant is one that includes nonlinear functions in its dynamics (eg. powers, sine, cosine).

## 5.1 Linearization

The easiest way to control a nonlinear system is by making it linear. The Jacobian matrix (2.2.3) can be calculated about the point.

> Beware that the accuracy of the linearized model decreases the farther away (in terms of state) you are calculating. If compute power allows,

## 5.2 Sliding Mode Control

# 6 State Estimation

## 6.1 Purpose

## 6.2 Observer

## 6.3 Measurement Model

## 6.4 Filtering

Filtering is the real-time subset of estimators.

### 6.4.1 *Calibration*

## 6.5 Complementary Filter

## 6.6 Alpha-Beta Filter

### 6.6.1 *Alpha-Beta-Gamma Filter*

## 6.7 Least-Squares Estimator

## 6.8 Linear-Quadratic Estimator (Kalman Filter)

### 6.8.1 *Extended Kalman Filter (EKF)*

The Kalman filter is limited to linear systems. One method to apply it to nonlinear systems is to linearize the system at the current time step, and perform the same algorithm.

### 6.8.2 *Unscented Kalman Filter (UKF)*

## 6.9 Particle Filter

### 6.9.1 *SIS Filter*

### 6.9.2 *SIR Filter*

### 6.9.3 *The Curse of Dimensionality*

## 6.10 Ensemble Kalman Filter (EnKF)

## 6.11 Sensor Fusion

### 6.11.1 *Madgwick Filter*

Note the quaternion derivative appears (Eq. 7.1).

### 6.11.2 *Mahoni Filter*

# 7 Modeling & Simulation

## 7.1 Reference Frames

### 7.1.1 *Inertial Frame*

### 7.1.2 *Body Frame*

### 7.1.3 *North-East-Down Frame*

### 7.1.4 *Intermediate Frames*

### 7.1.5 *Coriolis Forces*

## 7.2 Determining the State Vector

### 7.2.1 *Position & Derivatives*

### 7.2.2 *Rotation & Derivatives*

**Euler Angles**

**Direction Cosine Matrices**

**Quaternions**

**Quaternion Derivatives**

$$\dot{q} = \frac{1}{2}\Omega q, \quad \text{where } \Omega = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \tag{7.1}$$

$$\text{Recall that } \omega^b_{b/v} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

**Converting Between Representations** There are different methods based on which representation is being converted from/to. For ease, MATLAB has the built-in functions `eul2dcm`, `dcm2quat`, `quat2eul`. There is also the `angle2quat` function.

### 7.2.3 *Homogenous Transformation Matrix*

The idea behind the homogenous transformation matrix is to combine rotation and translation.

## 7.3 Flat Earth Equations of Motion

# 8 Optimal Control

## 8.1 Linear-Quadratic Regulator (LQR)

# 9 Robust Control

# 10 Adaptive Control

# 11 Predictive Control

# 12 Other Important Topics

## 12.1 Continuous Time vs Discrete Time

When the controller is calculated by a computer, the system and feedback cannot be evaluated in continuous time. The system can be thought of as proceeding in time-steps.

### 12.1.1 *Z Transform*

The Z transform is the corollary to the Laplace Transform, only for discrete time where Laplace is for continuous.

## 12.2 Optimization

### 12.2.1 *Gradient Descent*

## 12.3 Singular Value Decomposition

# 13 Appendix: Quick Reference

# 14 Appendix: MATLAB / Simulink Reference

This chapter contains practical methods / tutorials to accomplish tasks in MATLAB and Simulink

## 14.1 Linearization

### 14.1.1 *MATLAB*

### 14.1.2 *Simulink*

## 14.2 Animation

### 14.2.1 *MATLAB*

### 14.2.2 *Simulink*

# 15 Appendix: Python Reference

## 16 References