Rochester Institute of Technology

RIT Digital Institutional Repository

Theses

12-2013

# Sliding Mode Control of MIMO Non-Square Systems via Squaring Matrix Transforms

Tim Marvin

Follow this and additional works at: https://repository.rit.edu/theses

Recommended Citation

Marvin, Tim, "Sliding Mode Control of MIMO Non-Square Systems via Squaring Matrix Transforms" (2013). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the RIT Libraries. For more information, please contact repository@rit.edu.

# Sliding Mode Control of MIMO Non-Square Systems via Squaring Matrix Transforms

By

Tim Marvin

A Thesis Submitted in Partial Fulfillment of the Requirements
for Masters of Science in Mechanical Engineering

Approved by:

**Dr. Agamemnon Crassidis** – *Thesis Advisor* ______________________________________
Department of Mechanical Engineering

**Dr. Jason Kolodziej** ______________________________________
Department of Mechanical Engineering

**Dr. Wayne Walter** ______________________________________
Department of Mechanical Engineering

**Dr. Alan Nye** ______________________________________
Department of Mechanical Engineering

**Department of Mechanical Engineering**
**Rochester Institute of Technology**
**Rochester, New York 14623**
**December 2013**

**PERMISSION TO REPRODUCE THE THESIS**

# Sliding Mode Control of MIMO Non-Square Systems via Squaring Matrix Transforms

I, Tim Marvin, hereby grant permission to the Wallace Memorial Library of Rochester Institute of Technology to reproduce my thesis in the whole or part. Any reproduction will not be for commercial use or profit.

Date:_________________________ Signature:_____________________________

December 2013

# Abstract

In this thesis, a novel method for control of over- or under-actuated dynamic systems is developed. The primary control method considered here is Sliding Mode Control which requires an inversion of the control input influence matrix. However, on many systems this matrix is non-square, requiring alternate methods in order to obtain the control solution. Some existing solutions for this class of problems include pseudo-inversion such as the Moore-Penrose (which does not allow the design engineer to select the desired state to be controlled), dynamic extension (which is difficult to implement on large systems), and pseudo-inverse squaring transform methods. While the squaring transform method solves the key issue in the Moore-Penrose method of not being able to select the desired control state, it still has been limited to systems with only one input. The current effort seeks to extend this squaring transformation method to multiple input systems and demonstrate the control allocation properties of the technique. By extending this method to multiple-input systems the technique becomes applicable to a wider range of real world problems, allowing designers to select and optimally control any desired state on multi-input-multi-output systems. This thesis examines the existing solutions for squaring of input influence matrices such as Moore-Penrose and dynamic extension, the transform method developed in previous work, and derives a multi-input extension to that method and also considers control allocation in the solution process. Simulations are then developed on a two-input, four mass-spring-damper system, and a multi-input longitudinal aircraft model to demonstrate the technique and characterize its performance in both sterile and noisy environments. The results of these simulations demonstrate the significant performance gains in tracking performance, reduced control effort, and noise rejection compared to the legacy Moore-Penrose pseudo-inverse technique. In addition the ease of configuration for both desired state tracking and control allocation is demonstrated.

# Contents

## List of Figures

## List of Tables

## Nomenclature

| | |
|---|---|
| **A** | System Dynamic Matrix |
| **B** | Input Influence Matrix |
| $\mathbf{B}_{\dagger}$ | Moore-Penrose pseudo-inverse of **B** |
| **C** | Output Matrix |
| **x** | State Vector |
| **T** | Transformation Matrix |
| **T*** | Alternate form of Transformation Matrix |
| **Q** | State Weighting Matrix |
| **R** | Input Weighting Matrix |
| s | Sliding Surface |
| α | Positive Constant |
| γ | Positive Constant |
| λ | Positive Constant |
| **u** | Control Input |
| **y** | Transformed State Vector |
| J | Cost Function |
| **K**(t) | Solution to Linear Quadratic Regulator |
| **K** | Steady-State Solution to Linear Quadratic Regulator |
| $V_t$ | True Velocity |
| A | Angle of Attack |
| *p* | Roll Rate |
| *q* | Pitch Rate |
| *r* | Yaw Rate |
| *f* | Roll/Bank Angle |
| *θ* | Pitch Angle |
| *ψ* | Yaw/Heading Angle |
| V(x) | Lyapunov Function (of variable x) |
| $\mathbf{x}_{\mathbf{d}}$ | Subscript (d) denotes desired value (i.e. desired value of x) |
| $\tilde{\boldsymbol{x}}$ | Difference between x and $x_d$ (i.e. state error, x - $x_d$) |
| *n* | System Order/Number of States |
| *m* | Number of Inputs |
| *p* | Number of Outputs |
| s | Characteristic Root |

Note: Typical matrix notation is employed. A lowercase, italic variable indicates a scalar. A lowercase, bold variable indicates a vector. An uppercase, bold variable indicates a matrix.

# 1 Introduction

## 1.1 Background

### 1.1.1 Classical Control Theory and Methods

While classical control algorithms are widespread in industry today, they suffer from several drawbacks. Most are intended to be applied to only a Single Input and Single Output (SISO), and are unable to provide an optimal solution for many systems, since they don't incorporate a model of the plant. The shortfall can sometimes be improved through gain scheduling, but requires significant development and tuning on the designer's part. In addition to being limited to SISO systems, classical type controllers (of which PID is a common implementation) have performance drawbacks with non-linear systems since they ignore orders above 2 in the system response, and performance may be sensitive to modeling inaccuracies.

### 1.1.2 Modern Control Theory and Methods

Modern control theory has been progressing rapidly since the 1960's. It is characterized primarily by Multi Input and Multi Output (MIMO) systems, and operations in the time domain. Some advances in this area of research that are key components of this current effort include, the state-space modeling approach, Lyapunov's direct method, and sliding mode control.

#### *1.1.2.1 State-Space Models*

State space modeling provides a method to model a system in terms of a set of first-order differential equations. By replacing the higher order equations found in, for example, transfer function representations, with a set of first order equations, complex systems can be represented in a more compact form, as is most advantageous for systems with multiple inputs and outputs represented by a simple set of four matrices. [1] The generalized representation is usually of the form shown below:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) \tag{1}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}(t)\boldsymbol{x}(t) + \boldsymbol{D}(t)\boldsymbol{u}(t) \tag{2}$$

Where:

$$\begin{aligned}
\boldsymbol{x}(t) &= state\ vector \\
\boldsymbol{y}(t) &= output\ vector \\
\boldsymbol{u}(t) &= input\ vector \\
\boldsymbol{A}(t) &= system\ or\ state\ matrix \\
\boldsymbol{B}(t) &= control\ or\ input\ matrix \\
\boldsymbol{C}(t) &= output\ matrix \\
\boldsymbol{D}(t) &= feedthrough\ matrix \\
\dot{\boldsymbol{x}}(t) &= {}^{d}/_{dt}\,\boldsymbol{x}(t)
\end{aligned}$$

And:

$$\boldsymbol{A} \in \mathbb{R}^{n\;x\;n} \tag{3}$$

$$\boldsymbol{B} \in \mathbb{R}^{n\;x\;m} \tag{4}$$

$$\boldsymbol{C} \in \mathbb{R}^{r\;x\;n} \tag{5}$$

$$\boldsymbol{D} \in \mathbb{R}^{r\;x\;m} \tag{6}$$

With *n* equal to the number of states, *m* equal to the number of inputs, and *r* equal to the number of outputs.

In continuous time-invariant systems the **A**(t), **B**(t), **C**(t), and **D**(t) notation, simplifies to become just **A**, **B**, **C**, and **D** respectively.

By representing the system model in the compact state space form, the designer can more easily manage and manipulate the system. In addition, the system is less difficult to solve using numerical analysis tools since several numerical techniques existing for solving systems of first order differential equations.

As an example, a simple mass-spring-damper shown in Figure 1 is represented by its state space model in Equations (7) and (8). While the system in this example is simple, and so is the resulting state space model, complex models benefit immensely from the simple representation of first order differential equations during analysis and control design.

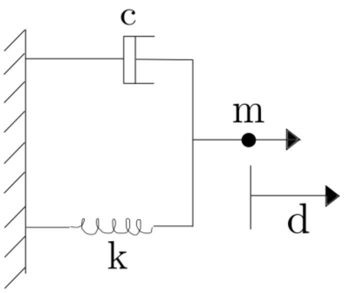

**Figure 1: Example Mass-Spring-Damper System (image courtesy Aaron Yoon)**

$$A = \begin{bmatrix} 0 & 1 \\ -k/m & 1/m \end{bmatrix} \tag{7}$$

$$B = \begin{bmatrix} 0 \\ 1/m \end{bmatrix} \tag{8}$$

### *1.1.2.2 Lyapunov Stability*

Aleksander Lyapunov developed a theory to determine stability of non-linear systems and published it in 1892. The theory found little use until the mid-1950's when the technique was applied to non-linear control systems (specifically Sliding Mode Controllers discussed below) [2]. The theory can be broken into several components or definitions which are shown below.

**Definition 1:** The equilibrium state x = 0 is said to be stable if, for any R > 0, there exists r > 0, such that if $\|x(0)\| < r$, then $\|x(t)\| < R$ for all of t ≥ 0. Otherwise the equilibrium point is unstable.

Definition 1 is the broadest in terms of limiting a system. It states that if a state trajectory is started at a point located within a radius r of an equilibrium point and the state remains within the radius R of the equilibrium point for all of t > 0 it is stable.

**Definition 2:** An equilibrium point 0 is asymptotically stable if it is stable, and if in addition there exists some r > 0 such that $\|x(0)\| < r$ implies that $x(t) \rightarrow 0$ as $t \rightarrow \infty$.

Definition 2 adds an additional restriction by defining the idea of asymptotic stability of a state. In this case if $x(t) \rightarrow 0$ as $t \rightarrow \infty$, the equilibrium point is said to be asymptotically stable.

**Definition 3:** An equilibrium point 0 is exponentially stable if there exists two strictly positive numbers α and λ such that

$$\forall t > 0, \|x(t)\| \leq \alpha \|x(0)\| e^{-\lambda t} \tag{9}$$

Definition 3 once again tightens the restrictions and states that for all times t > 0, that x(t) must not only approach zero, but that it must do it in an exponential fashion (its trajectory being bounded by $e^{-\lambda t}$ where λ is the rate of convergence).

**Definition 4:** A function, V(x), is said to be locally positive definite if V(0) = 0 and in a ball $B_{R0}$ $x \neq 0 \rightarrow V(x) > 0$. If V(0) = 0 and the above property holds over the entire state space, then V(x) is said to be globally positive definite.

**Definition 5:** If in a region defined by a ball $B_{R0}$, the function V(x) is positive definite and has continuous partial derivatives, and if its time derivative along any state trajectory of the system is negative semi-definite ($\dot{V}(x) \leq 0$) then V(x) is said to be a Lyapunov function for the given system.

Definition 4 accounted for the requirement that energy always be positive unless the state is 0, and Definition 5 accounts for the second requirement that energy always be decreasing. Stability is thus guaranteed since the functions are positive, and their derivatives are always moving towards the origin. These concepts will be applied during the development of example controllers.

#### 1.1.2.3 Sliding Mode Control

Sliding Mode Control was developed in the former USSR by several researchers [Aizerman and Gantmacher (1957), Emelyanov (1957), and Filippov (1960)] [2] during the 1950's and 60's and has continued to be an active area of research across several fields. [3] Some of its benefits include ease of application to a wide variety of typically difficult problems in controls such as nonlinear, MIMO, and large scale systems. Additionally, the method exhibits limited sensitivity to uncertainties (both intentionally un-modeled as a simplification, and un-known), and external disturbances assuming the bounds of these uncertainties are known. Along with these superior noise rejection capabilities, it also lends itself to state-space models providing the designer with a straightforward integration path. [3]

Drawbacks include the difficulty of inverting a non-square input influence matrix (also known as the B matrix). If the number system inputs do not match the system order, the matrix will be non-square. Non-square input influence matrices occur often in real world systems, especially in aircraft which commonly include redundant actuators and control paths. An additional drawback is the tendency of the controller to chatter, switching at a high rate, and potentially exciting undesired responses in the system under control. Chattering can be minimized, however, through various techniques such as a low pass filter on the output. [2]

As an example of a sliding mode controller consider the simple rotational system shown below: [3]

$$J\ddot{\theta}(t) = u(t) \tag{10}$$

Where:

$$\begin{gathered} J = inertia\ moment \\ \ddot{\theta}(t) = angle\ signal \\ u(t) = control\ input \end{gathered} \tag{11}$$

The sliding surface is defined as:

$$s(t) = cx(t) + \dot{x}(t) \tag{12}$$

Where:

$$c > 0 \tag{13}$$

$$x(t), \dot{x}(t) = state, and\ its\ derivative\ defining\ a\ phase\ plane \tag{14}$$

The tracking error function and the related derivatives are defined as:

$$x(t) = \theta(t) - \theta_d(t) \tag{15}$$

$$\dot{x}(t) = \dot{\theta}(t) - \dot{\theta}_d(t) \tag{16}$$

$$\ddot{x}(t) = \ddot{\theta}(t) - \ddot{\theta}_d(t) \tag{17}$$

Where θ, $\theta_d$, and their related derivatives are the actual and desired angular positions, velocities, and accelerations.

Taking the derivative of the sliding function (ensuring no movement of the state error trajectories is allowed once the trajectories reach the sliding surface) and substituting the error functions yield:

$$\dot{s}(t) = c\dot{x}(t) + \ddot{x}(t) \tag{18}$$

$$\dot{s}(t) = c\dot{x}(t) + \ddot{\theta}(t) - \ddot{\theta}_d(t) \tag{19}$$

$$\dot{s}(t) = c\left(\dot{\theta}(t) - \dot{\theta}_d(t)\right) + \ddot{\theta}(t) - \ddot{\theta}_d(t) \tag{20}$$

$$\dot{s}(t) = c\left(\dot{\theta}(t) - \dot{\theta}_d(t)\right) + \frac{1}{J}u(t) - \ddot{\theta}_d(t) \tag{21}$$

To ensure the Lyapunov stability (discussed in Section 1.1.2.2) of the function being positive, and the derivatives being negative, the following criteria must be met:

$$s\dot{s} < 0 \tag{22}$$

$$\frac{1}{2}s^2 \geq 0 \tag{23}$$

By substitution of the previous equations:

$$s\dot{s} = s\left[c\big(\dot{\theta} - \dot{\theta}_d\big) + \frac{1}{J}u - \ddot{\theta}_d\right] \tag{24}$$

Solving for **u**(t), the controller becomes:

$$\boldsymbol{u}(t) = J\big[c\big(\dot{\theta} - \dot{\theta}_d\big) + \ddot{\theta}_d - \eta sign(s)\big] \tag{25}$$

Where:

$$sign(s) = \begin{cases} 1, s > 0 \\ 0, s = 0 \\ -1, s < 0 \end{cases} \tag{26}$$

# 2 Background work

Combining state space models with sliding mode controllers provides the unique opportunity to handle MIMO control cases and achieve near perfect tracking of the selected states. Typically, however, the requirement exists that the input influence matrix be square (where the number of inputs is equal to the number of system states). Literature on the topic of dynamic inversion is available in a broad array of application areas, however, none make the novel leap to include control effort, MIMO systems, and control allocation in the derivation of the solution.

An example of aircraft applications include a dynamic inversion paper by Enns et al in 1994 [4] where the technique was applied to an F18 with the authors even noting the fact that the input influence matrix could be badly conditioned for the technique. The issue was avoided by not solving that case. Over time techniques have appeared expanding the range of problems and methods used such as in the more recent paper by Hameduddin in 2011 [5] which applied the method to aircraft trajectory tracking. Hameduddin et al relied on the Moore-Penrose (discussed in Section 3.1.1) inverse to address the non-square input influence matrix of a MIMO system, and was thus unable to take into account control effort or state tracking selection. Aside from the papers of [6] [7] who themselves expressed the lack of documented work in using a technique such as this, there is little references in the transformation matrix technique discussed herein.

## 2.1 Non-Square Systems

In real world systems it is exceedingly rare to find a system in which the number of system inputs matches the system order. As mentioned previously, systems such as aircraft (both high performance and commercial) typically include redundant control actuators. The situation alone can create a case where the number of system states exceeds the number of system inputs. In these types of scenarios (and others, as both under- and over-actuated systems exhibit this issue) an inversion of a non-square B matrix to derive the Sliding Mode Control solution is required.

### 2.1.1 Moore-Penrose Pseudo-inverse

Mathematical solutions for inverting non-square matrices do exist in common practice and domain specific literature, however none address all of the desired requirements when inverting to implement a Sliding Mode Controller. The most common solution used in literature is the Moore-Penrose pseudo-inverse. [8] [9] While this solution does share some of the properties of a true inverse and can be used in the implementation of a Sliding Mode Controller, the suboptimal technique does not provide a method to target desired states for tracking purposes, and/or minimize cost in terms of controller effort. In fact its tracking ability is limited to the actuated states only. [7]

### 2.1.2 Dynamic Extension

Dynamic extension is another technique capable of modifying a system such that the input influence B matrix is squared through a transformation. While the method is capable of generating a square B matrix and thus allowing perfect tracking via SMC, the level of effort for calculating the dynamic extension becomes impractical for systems exceeding 2 or 3 states and likely explains its rarity in published literature [2] .

### 2.1.3 Current Effort – Squaring Transformation Matrix

Most recent solutions in the development of a squaring transformation matrix technique were developed by Schkoda, DiFiore, and Crassidis; with extension to MIMO systems. Schkoda's research was concerned with the development of a controller for non-square SISO under actuated systems. DiFiore's work used the case of actuators being equal to the number of states to collapse the problem to a SISO solution. While these developments were progressions towards applications on some real world systems (leaving behind pseudo-inverse techniques and applying it to fuel cell control problem) the existing solutions did not address the case of multiple system inputs where actuators do not match the state count. This limits the applicability of this technique on real world systems (e.g. aircraft which implement redundant actuators) until the extension is made. Both Schkoda and DiFiore proposed future efforts in developing the extension attempted herein.

To develop and demonstrate the extension to MIMIO systems the following steps were taken:

1. Research existing methods to understand the theoretical fundamentals of the square and under-actuated dynamic inversion solutions
2. Examine the theoretical solution for under-actuated systems by mathematically extending existing techniques to MIMO problems while retaining the control allocation and effort considerations
3. Simulate the dynamic response of various example systems operating under the proposed control law. (A multiple input mass spring damper and longitudinal aircraft model)

# 3 Theoretical Development

## 3.1 Squaring Transformation Matrix for Under Actuated Systems

None of above techniques includes a cost function into their computations making them difficult to implement directly in real world applications allowing for control allocation and tracking of selectable state. For example the current solutions do not take into account the range of an actuator, control surface loading, and available power; nor do current solutions attempt to minimize the control displacement and its impact on rate and position saturation. [6] Additionally, the Moore-Penrose pseudo inverse does not allow for control allocation of system actuators at all.

Schkoda, whose paper and thesis were the inspiration for this research, developed a method to compute a transformation matrix to ‘square up’ the input influence matrix for the purposes of solving the sliding mode controller equations. The initial derivation is briefly summarized below. [6]

Given a system defined by:

$$\dot{x} = \boldsymbol{Ax} + \boldsymbol{Bu} \tag{27}$$

Where:

$$\boldsymbol{A} \in \mathbb{R}^{n\,x\,n} \tag{28}$$

$$\boldsymbol{B} \in \mathbb{R}^{n\,x\,1} \tag{29}$$

Note that **B** is non-square in the case of an under- or over-actuated system.

Define the sliding surface as:

$$s = \boldsymbol{x} - \boldsymbol{x_d} + \gamma \int_0^t (\boldsymbol{x} - \boldsymbol{x_d})dr \tag{30}$$

Take the derivative and set the result equal to 0:

$$\dot{s} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{\boldsymbol{d}} + \gamma\tilde{\boldsymbol{x}} = 0 \tag{31}$$

Substituting the original state space model into this equation yields:

$$0 = A\boldsymbol{x} + B\boldsymbol{u} - \dot{\boldsymbol{x}}_{\boldsymbol{d}} + \gamma\tilde{\boldsymbol{x}} \tag{32}$$

$$-\boldsymbol{A}x + \dot{x}_d - \gamma\tilde{x} = \boldsymbol{Bu} \tag{33}$$

$$\boldsymbol{u} = \boldsymbol{B}^{-1}[\dot{x}_d - \boldsymbol{Ax} - \gamma\tilde{x}] \tag{34}$$

Therefore the input influence matrix (or **B** matrix) must be inverted to solve for the control effort. Schkoda [6] proposed a coordinate transformation such that the transformed system would result in a square (and thus invertible) system by defining the following transformation:

$$y = \boldsymbol{T}x \tag{35}$$

Where:

$$\boldsymbol{T} \in \mathbb{R}^{1\,x\,n} \tag{36}$$

Note that the dimensions of **T** are the transpose of **B**.

Differentiating the transformation and substituting it into the original state space equation yields:

$$\dot{\boldsymbol{y}} = \boldsymbol{T}\dot{\boldsymbol{x}} \rightarrow \dot{\boldsymbol{x}} = \boldsymbol{A}x + \boldsymbol{B}u \tag{37}$$

$$\dot{\boldsymbol{y}} = \boldsymbol{T}[\boldsymbol{Ax} + \boldsymbol{Bu}] = \boldsymbol{TAx} + \boldsymbol{TBu} \tag{38}$$

A new sliding surface is defined as:

$$\boldsymbol{s} = \boldsymbol{y} - \boldsymbol{y_d} + \gamma\int_0^t (\boldsymbol{y} - \boldsymbol{y_d})dr \tag{39}$$

Differentiating the sliding surface and setting it equal to zero once again (utilizing Leibniz’s Rule [10] which allows us to move the differential through the integral insert the limits):

$$\dot{\boldsymbol{s}} = \dot{\boldsymbol{y}} - \dot{\boldsymbol{y}}_{\boldsymbol{d}} + \gamma\tilde{\boldsymbol{y}} = 0 \tag{40}$$

Substituting our updated state space equation into our sliding surface equation yields:

$$\boldsymbol{TAx} + \boldsymbol{TBu} - \dot{\boldsymbol{y}}_{\boldsymbol{d}} + \gamma\tilde{\boldsymbol{y}} = 0 \tag{41}$$

$$\boldsymbol{TB}u = \dot{\boldsymbol{y}}_{\boldsymbol{d}} - \boldsymbol{TA}x - \gamma\tilde{\boldsymbol{y}} = 0 \tag{42}$$

$$\boldsymbol{u} = (\boldsymbol{TB})^{-1}[\dot{\boldsymbol{y}}_{\boldsymbol{d}} - \boldsymbol{TA}x - \gamma\tilde{\boldsymbol{y}}] \tag{43}$$

Note again that the choice of **T** will result in **TB** being square, thus as mentioned above:

$$\boldsymbol{T} \in \mathbb{R}^{m\,x\,n} \tag{44}$$

The problem now reduces to finding **T** such that its dimensions are the transpose of **B**'s, where the tracking of selectable states is possible, and minimization of the control effort required. A possible solution is to employ a Linear Quadratic Regulator (LQR) cost function used extensively in optimal control problems to find an "optimal" T for the selection of desirable tracking states and including allocation of the control effort:

$$J = \frac{1}{2}\boldsymbol{x}^T(t_f)S\boldsymbol{x}(t_f) + \frac{1}{2}\int_{t_0}^{t_f}(x^T\boldsymbol{Q}x + u^T\boldsymbol{R}\boldsymbol{u})dt \tag{45}$$

**Q** is chosen to weight the states the designer desires to track, and **R** is used to allocate the control effort. The standard LQR solution in optimal control has the following control feedback form for the control effort:

$$\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x} \tag{46}$$

Substitute the previous equation into the following and solve for **T**:

$$\boldsymbol{u} = -(\boldsymbol{TB})^{-1}\boldsymbol{T}[\boldsymbol{A} + \gamma\boldsymbol{I}]\boldsymbol{x} \tag{47}$$

$$\boldsymbol{K} = (\boldsymbol{TB})^{-1}\boldsymbol{T}[\boldsymbol{A} + \gamma\boldsymbol{I}] \tag{48}$$

$$\boldsymbol{T} = (\boldsymbol{T}_*\boldsymbol{B})\boldsymbol{K}(\boldsymbol{A} + \gamma\boldsymbol{I})^{-1} \tag{49}$$

Where:

$$\boldsymbol{T}_* = \boldsymbol{B}^T \tag{50}$$

Substituting T back into our control law equation yields:

$$\boldsymbol{u} = (\boldsymbol{T}_*\boldsymbol{B})^{-1}\boldsymbol{T}[\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{A}\boldsymbol{x} - \gamma\tilde{\boldsymbol{x}}] \tag{51}$$

While this technique does solve the non-square matrix problem, and allows tracking of specific states, it was only derived for one input to the system. DiFiore started to extend the method to multiple inputs however the number of actuators matched the number of states and thus the problem reduced to a SISO type system once again. [7] As demonstrated in the example below, the proposed technique can be extended to under-actuated systems with multiple inputs while maintaining the control allocation and effort benefits.

## 3.2 MIMO Derivation

Below is the re-derivation of the above technique to address MIMO systems.

Given a system defined by:

$$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu} \tag{52}$$

Where:

$$\boldsymbol{A} \in \mathbb{R}^{n\,x\,n} \tag{53}$$

$$\boldsymbol{B} \in \mathbb{R}^{n\,x\,m} \tag{54}$$

Note **B** is defined to be non-square as in the case of an under- or over-actuated system and in this case *m* is greater than 1.

Define the sliding surface as before:

$$s = \boldsymbol{x} - \boldsymbol{x_d} + \gamma \int_0^t (\boldsymbol{x} - \boldsymbol{x_d}) dr \tag{55}$$

Take the derivative and setting the result equal to zero ensures no movement of the state tracking error dynamics once the trajectories reach the sliding surface and yields:

$$\dot{\boldsymbol{s}} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x_d}} + \gamma\widetilde{\boldsymbol{x}} = 0 \tag{56}$$

Inputting the state space model into the SMC equations yields:

$$0 = \boldsymbol{Ax} + \boldsymbol{Bu} - \dot{\boldsymbol{x}}_{\boldsymbol{d}} + \gamma\widetilde{\boldsymbol{x}} \tag{57}$$

$$-\boldsymbol{Ax} + \dot{\boldsymbol{x}}_{\boldsymbol{d}} - \gamma\widetilde{\boldsymbol{x}} = \boldsymbol{Bu} \tag{58}$$

$$\boldsymbol{u} = \boldsymbol{B}^{-1}[\dot{\boldsymbol{x}}_{\boldsymbol{d}} - A\boldsymbol{x} - \gamma\widetilde{\boldsymbol{x}}] \tag{59}$$

Once again, as anticipated; the inversion of the input influence **B** matrix is required. Defining a coordinate transformation as follows:

$$\boldsymbol{y} = \boldsymbol{Tx} \tag{60}$$

Where:

$$\boldsymbol{T} \in \mathbb{R}^{m\,x\,n} \tag{61}$$

Noting that the dimensions of **T** are the transpose of **B**, and m is greater than 1.

Differentiating **T** and substituting it into the original state space equation yields:

$$\dot{\boldsymbol{y}} = \boldsymbol{T}\dot{\boldsymbol{x}} \rightarrow \dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu} \tag{62}$$

$$\dot{\boldsymbol{y}} = \boldsymbol{T}[\boldsymbol{Ax} + \boldsymbol{Bu}] = \boldsymbol{TAx} + \boldsymbol{TBu} \tag{63}$$

A new sliding surface can be defined as:

$$\boldsymbol{s} = \boldsymbol{y} - \boldsymbol{y_d} + \gamma \int_0^t (\boldsymbol{y} - \boldsymbol{y_d}) dr \tag{64}$$

Differentiating Eq. (64) and setting it equal to zero once again (utilizing Leibniz's Rule [10]):

$$\dot{\boldsymbol{s}} = \dot{\boldsymbol{y}} - \dot{\boldsymbol{y}}_{\boldsymbol{d}} + \gamma\widetilde{\boldsymbol{y}} = 0 \tag{65}$$

Finally, inserting the new state space equation into the sliding surface equation yields:

$$\boldsymbol{TAx} + \boldsymbol{TBu} - \dot{\boldsymbol{y}}_{\boldsymbol{d}} + \gamma\widetilde{\boldsymbol{y}} = 0 \tag{66}$$

$$\boldsymbol{TB}u = \dot{\boldsymbol{y}}_{\boldsymbol{d}} - \boldsymbol{TAx} - \gamma\widetilde{\boldsymbol{y}} = 0 \tag{67}$$

$$\boldsymbol{u} = (\boldsymbol{TB})^{-1}[\dot{\boldsymbol{y}}_{\boldsymbol{d}} - \boldsymbol{TAx} - \gamma\widetilde{\boldsymbol{y}}] \tag{68}$$

Note: With **B** having *m* greater than 1, the product of **TB** will remain square due to the selection of **T** as the transpose of **B**:

$$\boldsymbol{T} \in \mathbb{R}^{m\,x\,n} \tag{69}$$

The problem now reduces to finding a **T** such that the dimensions of **T** are the transpose of **B**'s, near perfect tracking at desired selectable states is achieved, and control effort is minimized through control allocation. Choosing **T** such that the LRQ cost function is minimized (assuming a linearized plant):

$$J = \frac{1}{2}x^T(t_f)Sx(t_f) + \frac{1}{2}\int_{t_0}^{t_f}(x^T\boldsymbol{Q}x + u^T\boldsymbol{R}u)dt \tag{70}$$

Choosing **Q** in such that the desired states to track are weighted, and **R** to allocate the control effort, the standard LQR problem can be solved for the feedback gain matrix **K**:

$$\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x} \tag{71}$$

Substituting into our control equation and solving for **T**:

$$\boldsymbol{u} = -(\boldsymbol{T}\boldsymbol{B})^{-1}\boldsymbol{T}[\boldsymbol{A} + \gamma\boldsymbol{I}]\boldsymbol{x} \tag{72}$$

$$\boldsymbol{K} = (\boldsymbol{T}\boldsymbol{B})^{-1}\boldsymbol{T}[\boldsymbol{A} + \gamma\boldsymbol{I}] \tag{73}$$

$$\boldsymbol{T} = (\boldsymbol{T}_*\boldsymbol{B})\boldsymbol{K}(\boldsymbol{A} + \gamma\boldsymbol{I})^{-1} \tag{74}$$

Where:

$$\boldsymbol{T}_* = \boldsymbol{B}^T \tag{75}$$

Substituting ***T*** back into our original control law equation:

$$\boldsymbol{u} = (\boldsymbol{T}_*\boldsymbol{B})^{-1}\boldsymbol{T}[\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{A}\boldsymbol{x} - \gamma\tilde{\boldsymbol{x}}] \tag{76}$$

The resulting control law has the properties of a sliding mode controller (noise rejection, and stability assurance) with the addition control allocation and the minimization of control effort.

# 4 Results

To demonstrate the effectiveness of the newly derived control technique, several simulations were developed to model the system responses, and quantify the resulting performance. The Moore-Penrose inverse was used as the baseline or legacy technique. These simulations were performed on a four-mass-spring-damper problem and a longitudinal aircraft model. The primary control objective undertaken in these simulations was, given a set of desired states; track one of the non-directly actuated states, with the minimum control effort.

## 4.1 Under-Actuated System Example

To demonstrate the technique a derivation and results of the technique discussed previously the method is applied to a basic 4 mass-spring-damper system with two force inputs. A diagram showing the system is presented in Figure 2 below.

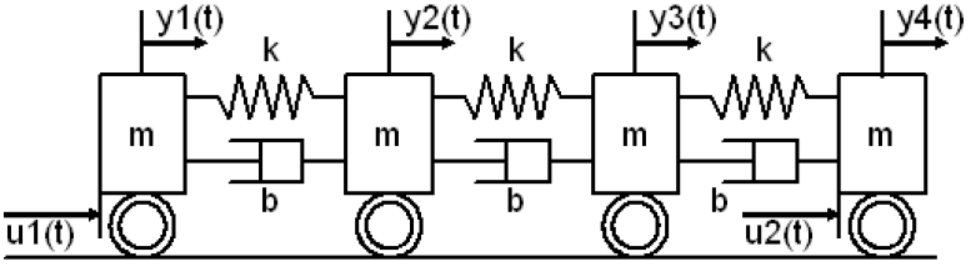

Figure 2: Four Mass-Spring-Damper System (image courtesy Markus Kottmann)

### 4.1.1 System Assumptions

Input forces:

$$\boldsymbol{u} = [u_1 u_2] \tag{77}$$

Four positions and four velocities as outputs (which match the internal states below):

$$\boldsymbol{y} = [y_1\ y_2\ y_3\ y_4\ \dot{y}_1\ \dot{y}_2\ \dot{y}_3\ \dot{y}_4] \tag{78}$$

State variables are assumed to be:

$$\begin{aligned} x_1 &= y_1 \\ x_2 &= y_2 \\ x_3 &= y_3 \\ x_4 &= y_4 \\ x_5 &= \dot{y}_1 \\ x_6 &= \dot{y}_2 \end{aligned} \tag{79}$$

$$x_7 = \dot{y}_3$$
$$x_8 = \dot{y}_4$$

Physical constants are given as:

$$\begin{aligned} m &= 1kg \\ k &= 36\frac{N}{m} \\ b &= 0.6\frac{N \cdot s}{m} \end{aligned} \tag{80}$$

### 4.1.2 State Space Model Derivation

Starting from the system model shown in Figure 2 above, the state space model was derived in the process shown below.

For reference the system's free body diagrams are shown in Figure 3:

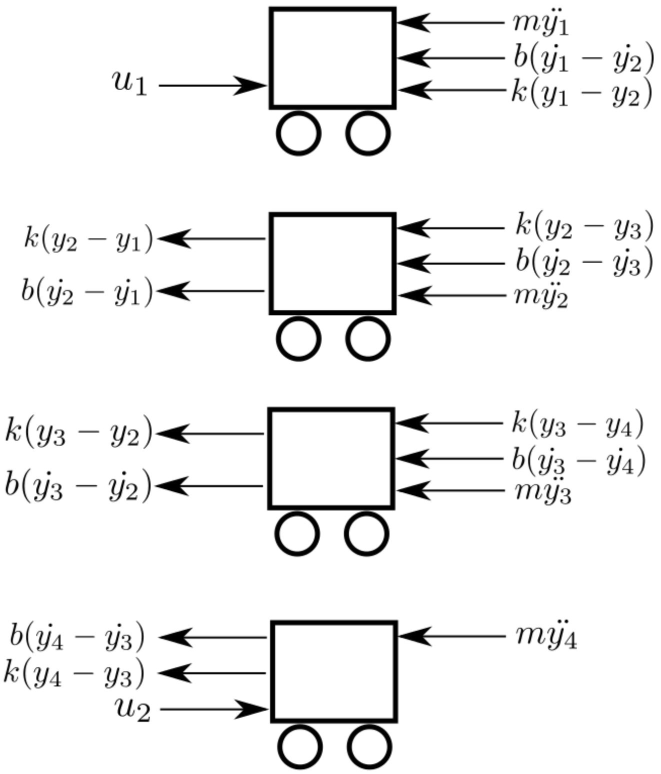
$m\ddot{y}_1$
$b(\dot{y}_1 - \dot{y}_2)$
$u_1$
$k(y_1 - y_2)$
$k(y_2 - y_3)$
$k(y_2 - y_1)$
$b(\dot{y}_2 - \dot{y}_3)$
$b(\dot{y}_2 - \dot{y}_1)$
$m\ddot{y}_2$
$k(y_3 - y_4)$
$k(y_3 - y_2)$
$b(\dot{y}_3 - \dot{y}_4)$
$b(\dot{y}_3 - \dot{y}_2)$
$m\ddot{y}_3$
$b(\dot{y}_4 - \dot{y}_3)$
$m\ddot{y}_4$
$k(y_4 - y_3)$
$u_2$

**Figure 3: Free-body diagrams for four mass-spring-damper system**

Using Lagrange's energy method to compute the equations of motion the system is described via a set of energy equations [11].

With the kinetic energy elements of the system represented by T:

$$T = \frac{1}{2}m\dot{y}_1^2 + \frac{1}{2}m\dot{y}_2^2 + \frac{1}{2}m\dot{y}_3^2 + \frac{1}{2}m\dot{y}_4^2 \tag{81}$$

And the potential energy elements represented by V:

$$V = \frac{1}{2}k(y_2 - y_1)^2 + \frac{1}{2}k(y_3 - y_2)^2 + \frac{1}{2}k(y_4 - y_3)^2 \tag{82}$$

Solving for the Lagrangian representation, denoted by L:

$$L = T - V \tag{83}$$

$$L = \frac{1}{2}m\dot{y}_1^2 + \frac{1}{2}m\dot{y}_2^2 + \frac{1}{2}m\dot{y}_3^2 + \frac{1}{2}m\dot{y}_4^2 \tag{84}$$

$$-\frac{1}{2}k(y_2-y_1)^2-\frac{1}{2}k(y_3-y_2)^2-\frac{1}{2}k(y_4-y_3)^2$$

Computing the Rayleigh dissipation elements, denoted by R:

$$R=\frac{1}{2}b(\dot{y_2}-\dot{y_1})^2+\frac{1}{2}b(\dot{y_3}-\dot{y_2})^2+\frac{1}{2}b(\dot{y_4}-\dot{y_3})^2 \tag{85}$$

Given the Lagrange equation below:

$$F_{q_i}=\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q_\imath}}\right]-\frac{\partial L}{\partial q_i}+\frac{\partial R}{\partial \dot{q_\imath}} \tag{86}$$

By substitution and differentiation, the four equations of motion are found to be:

$$m\ddot{y_1}-k(y_2-y_1)-b(\dot{y_2}-\dot{y_1})=u_1 \tag{87}$$

$$m\ddot{y_2}+k(y_2-y_1)-k(y_3-y_2)+b(\dot{y_2}-\dot{y_1})-b(\dot{y_3}-\dot{y_2})=0 \tag{88}$$

$$m\ddot{y_3}+k(y_3-y_2)-k(y_4-y_3)+b(\dot{y_3}-\dot{y_2})-b(\dot{y_4}-\dot{y_3})=0 \tag{89}$$

$$m\ddot{y_4}+k(y_4-y_3)+b(\dot{y_4}-\dot{y_3})=u_2 \tag{90}$$

Solve for the eight state equations:

$$\dot{x_1}=\dot{y_1}=x_5 \tag{91}$$

$$\dot{x_2}=\dot{y_2}=x_6 \tag{92}$$

$$\dot{x_3}=\dot{y_3}=x_7 \tag{93}$$

$$\dot{x_4}=\dot{y_4}=x_8 \tag{94}$$

$$\dot{x_5}=\ddot{y_1}=-\frac{b}{m}(\dot{y_1}-\dot{y_2})-\frac{k}{m}(y_1-y_2)+\frac{u_1}{m} \tag{95}$$

$$\dot{x_6}=\ddot{y_2}=-\frac{k}{m}(y_2-y_1)-\frac{b}{m}(\dot{y_2}-\dot{y_1})-\frac{k}{m}(y_2-y_3)-\frac{b}{m}(\dot{y_2}-\dot{y_3}) \tag{96}$$

$$\dot{x_7}=\ddot{y_3}=-\frac{b}{m}(\dot{y_3}-\dot{y_2})-\frac{k}{m}(y_3-y_2)-\frac{b}{m}(\dot{y_3}-\dot{y_4})-\frac{k}{m}(y_3-y_4) \tag{97}$$

$$\dot{x_8}=\ddot{y_4}=-\frac{b}{m}(\dot{y_4}-\dot{y_3})-\frac{k}{m}(y_4-y_3)+\frac{u_2}{m} \tag{98}$$

Rearranging yields:

$$\dot{x_5} = \frac{1}{m}[-kx_1 + kx_2 - bx_5 + bx_6 + u_1] \tag{99}$$

$$\dot{x_6} = \frac{1}{m}[kx_1 - kx_2 - kx_2 + kx_3 + bx_5 - bx_6 - bx_6 + bx_7] \tag{100}$$

$$\dot{x_7} = \frac{1}{m}[kx_2 - kx_3 - kx_3 + kx_4 + bx_6 - bx_7 - bx_7 + bx_8] \tag{101}$$

$$\dot{x_8} = \frac{1}{m}[kx_3 - kx_4 + bx_7 - bx_8 + u_2] \tag{102}$$

Formatting the above state equations into standard state space matrix format yields:

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{k}{m} & \frac{k}{m} & 0 & 0 & -\frac{b}{m} & \frac{b}{m} & 0 & 0 \\ \frac{k}{m} & -2*\frac{k}{m} & \frac{k}{m} & 0 & \frac{b}{m} & -2*\frac{b}{m} & \frac{b}{m} & 0 \\ 0 & \frac{k}{m} & -2*\frac{k}{m} & \frac{k}{m} & 0 & \frac{b}{m} & -2*\frac{b}{m} & \frac{b}{m} \\ 0 & 0 & \frac{k}{m} & -\frac{k}{m} & 0 & 0 & \frac{b}{m} & -\frac{b}{m} \end{bmatrix} \tag{103}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m} \end{bmatrix} \tag{104}$$

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{105}$$

$$\boldsymbol{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{106}$$

### 4.1.3 Pseudo-Inverse Controller

Given the state space system defined previously in Equations (103) – (106) of the form:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{107}$$

Where:

$$\boldsymbol{A} \in \mathbb{R}^{8\,x\,8} \tag{108}$$

$$\boldsymbol{B} \in \mathbb{R}^{8\,x\,2} \tag{109}$$

Note: **B** is defined to be non-square as in the case of an under- or over-actuated system and in this case *m* is 2.

Define the sliding surface as before (see Equation (30)):

$$s = \boldsymbol{x} - \boldsymbol{x_d} + \gamma \int_0^t (\boldsymbol{x} - \boldsymbol{x_d})dr \tag{110}$$

Take the derivative and setting the result equal to zero yields:

$$\dot{\boldsymbol{s}} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x_d}} + \gamma\widetilde{\boldsymbol{x}} = 0 \tag{111}$$

Inputting the state space model into the SMC equations yields:

$$0 = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} - \dot{\boldsymbol{x}}_{\boldsymbol{d}} + \gamma\widetilde{\boldsymbol{x}} \tag{112}$$

$$-\boldsymbol{A}\boldsymbol{x} + \dot{\boldsymbol{x}}_{\boldsymbol{d}} - \gamma\widetilde{\boldsymbol{x}} = \boldsymbol{B}\boldsymbol{u} \tag{113}$$

$$\boldsymbol{u} = \boldsymbol{B}^{-1}[\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{A}\boldsymbol{x} - \gamma\widetilde{\boldsymbol{x}}] \tag{114}$$

Solving our final control law seen in Equation (114) now requires the inverse of B. B is of dimensions 8 x 2.

The MATLAB implementation of the controller is formed as shown in Equation (115):

$$\boldsymbol{u} = -pinverse(\boldsymbol{B}) * [(\boldsymbol{A} + \gamma * \boldsymbol{I})\boldsymbol{x} - (\gamma * \boldsymbol{x_d}) - \dot{\boldsymbol{x}}_{\boldsymbol{d}}] \tag{115}$$

The controller is then realized in the Simulink model as shown in Figure 4:

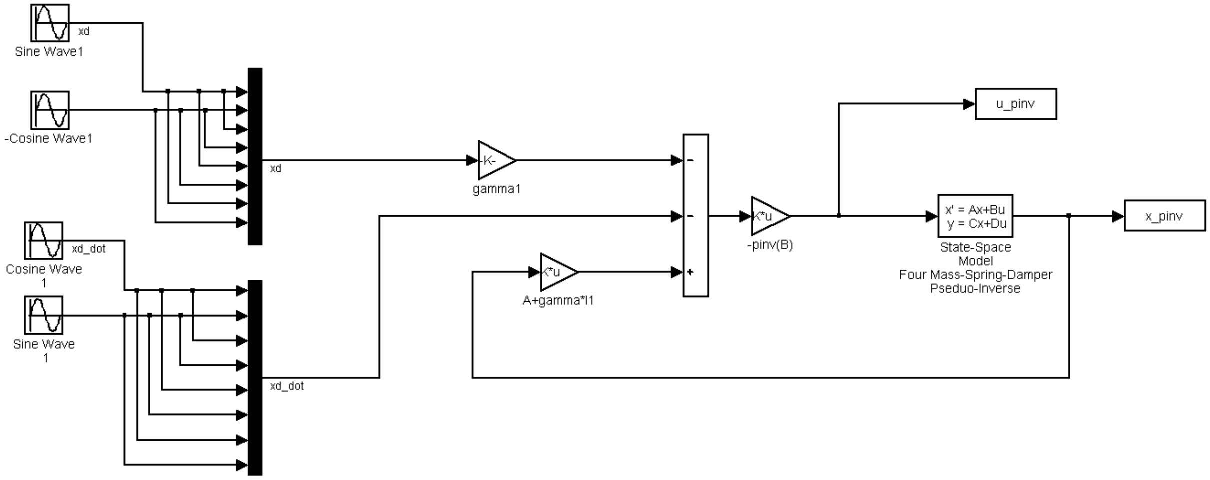

**Figure 4: Four Mass Spring Damper Simulink Pseudo-Inverse Controller Model**

### 4.1.4 Transformed Matrix Controller

Where:

$$\boldsymbol{T}_{*} = \boldsymbol{B}^{T} \tag{116}$$

Solving for **T** based on the derivation shown in Section 3.2 above. **K** is the gain matrix found by solving the standard LQR problem with the following inputs. Parameter gamma and matrix **R** are selected by starting with values used in previous research papers in this area, in combination order of magnitude LQR parameter selection techniques. **Q** is selected such that the desired control allocation state is several orders of magnitude above the remaining states. In the current example, it can be seen that to track Cart 3's position **Q**[3,3] is set to 1,000,000,000, while the remainder of the matrix is the identity matrix. To track Cart 2's position, **Q**[2,2] would be set similarly. This selection may be made for any of the 8 available states.

Setting **Q** to track the desired state:

$$\boldsymbol{Q}_{Cart2Position} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000000000 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{117}$$

$$\boldsymbol{Q}_{Cart3Position} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000000000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{118}$$

$$\boldsymbol{R} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \tag{119}$$

$$\gamma = 20 \tag{120}$$

Substitute the resulting **K** (feedback gain matrix) obtained via MATLAB’s `lqr()`function into the earlier derived formula for **T**:

$$\boldsymbol{T} = \boldsymbol{T}_{*}\boldsymbol{B}\boldsymbol{K}[\boldsymbol{A} + \gamma * eye(8)]^{-1}$$

$$\xrightarrow{yields}$$

$$\boldsymbol{T} = \begin{bmatrix} 0.6738 & -1.3278 & -1.0004 & -0.8526 & 0.1536 & 0.0422 & -0.2322 & 0.0833 \\ 0.5329 & 34.9241 & 429.1039 & 35.4328 & -0.0169 & 0.6982 & 18.8197 & 0.6807 \end{bmatrix} \tag{121}$$

The following equation is realized in Simulink as shown in Figure 5:

$$u = -inv(\boldsymbol{T}_{*} * \boldsymbol{B})\boldsymbol{T} * [(\boldsymbol{A} + \gamma * \boldsymbol{I})x - (\gamma * x_d) - \dot{x_d}] \tag{122}$$

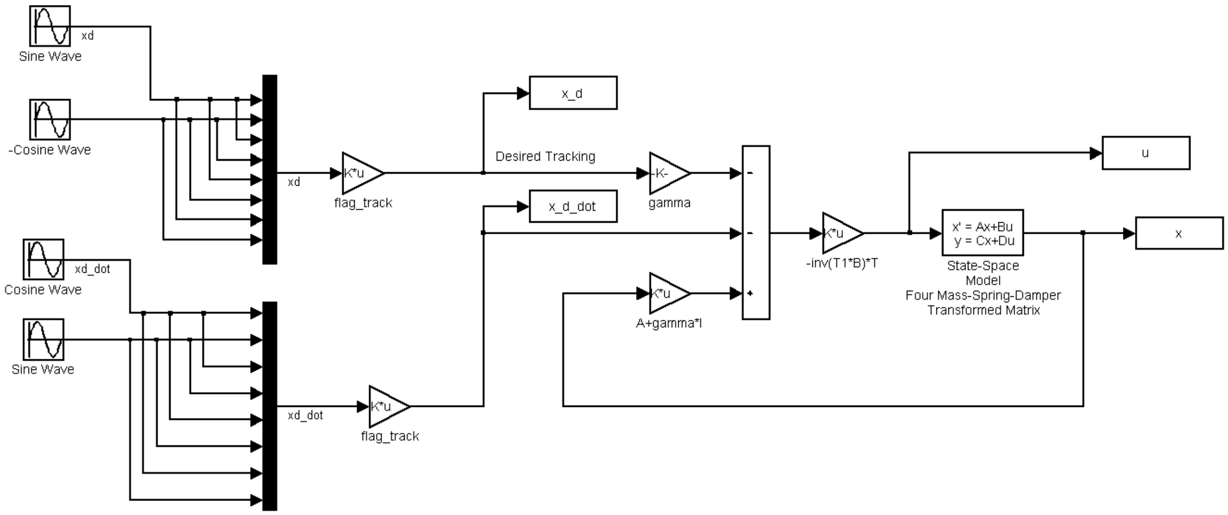

**Figure 5: Four Mass Spring Damper Transformed Matrix Simulink Model**

### 4.1.5 Results and Comparison of Techniques

Given the Simulink model described in Section 4.1.4, numerical simulations were ran to establish the system response and controller performance for both discussed techniques. In these simulations, a target state was selected to optimize tracking of. While all states were simulated, a representative example, State Three (Cart Three's position), is examined here. Shown in Figure 6 (and Figure 8), the tracking of State Three was nearly perfect and State Four had minimal error when controlled with the transformed matrix technique. While this transformed matrix control methodology did not exhibit particularly good performance in tracking States One and Two, they were not targeted during the system design so this is to be expected. The tracking of these non-primary states can likely be tuned via optimization of Q matrix value selection.

In comparison, the performance of the Moore-Penrose pseudo-inverse controller, shown in Figure 7 (and Figure 8), was quite poor on the selected State Three in tracking accuracy. In fact its tracking was nearly complete out of phase of the desired position (State Three), and the other states exhibited similarly poor tracking. Overall its tracking was poor on all states except the two velocity states, which were nearly perfect. This response is to be expected as Moore-Penrose techniques track only the directly connected states well.

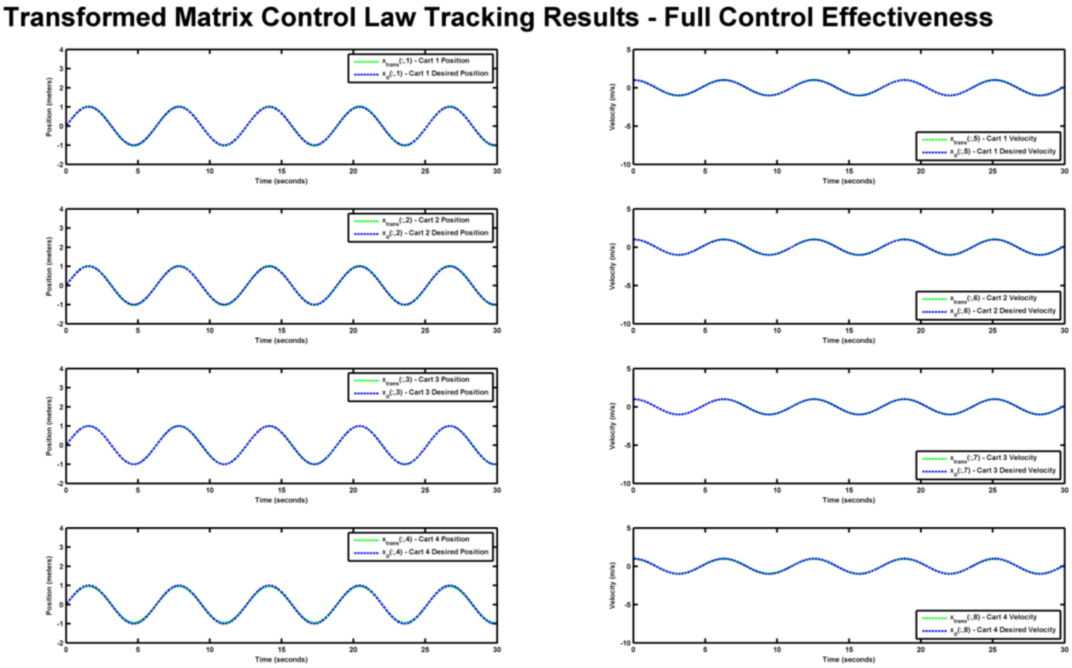

Figure 6: Transformed Matrix Tracking Results

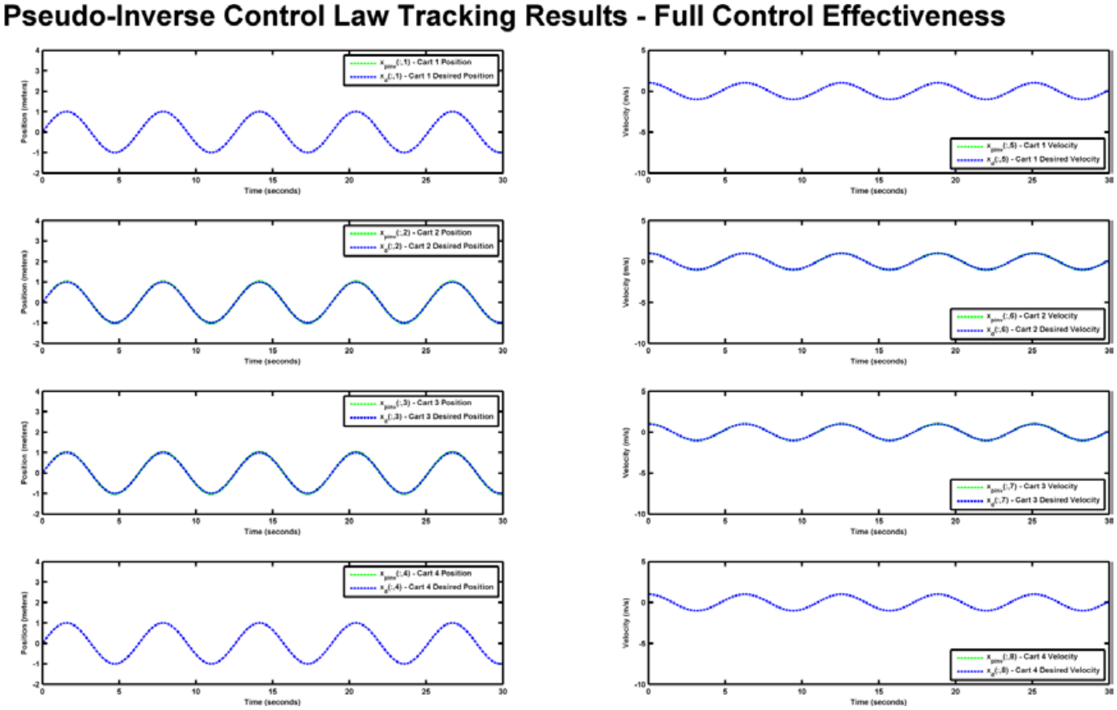

Figure 7: Pseudo-Inverse Tracking Results

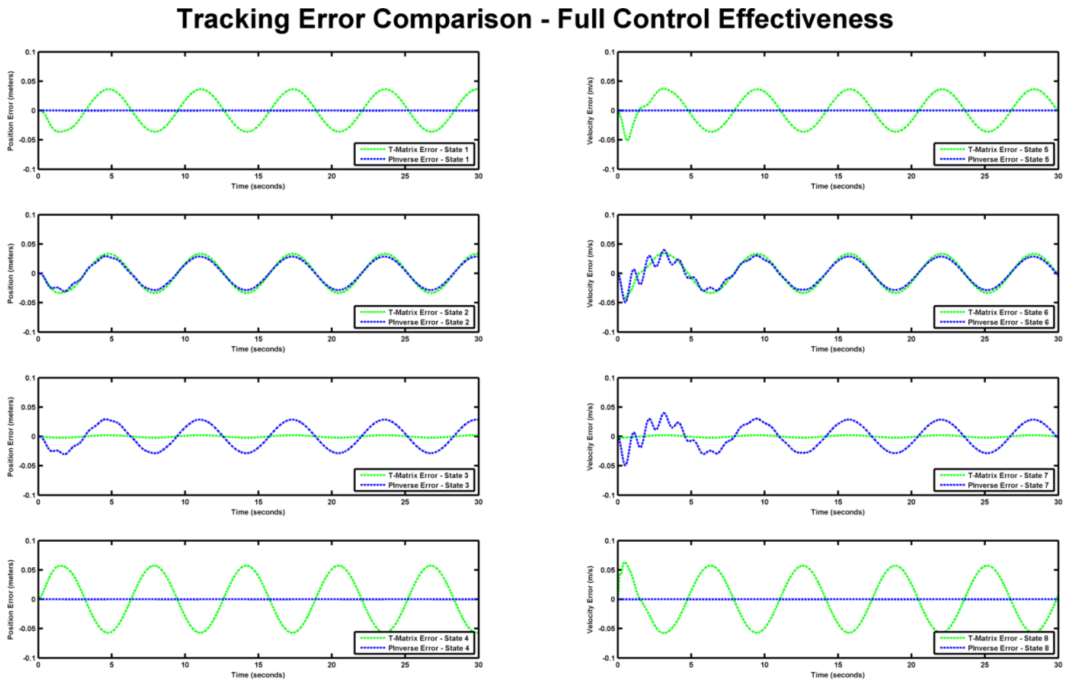

Figure 8: Tracking Error

The control effort comparison between the techniques is shown in Figure 9 below. It is apparent that the transformed matrix technique requires an initial spike in control effort to start system movement, but returns to a level comparable to that of the Moore-Penrose method immediately. While the control effort is comparable after the initial startup spike, as previously shown the Moore-Penrose exhibits poor tracking accuracy throughout and thereby its primary disadvantage; the inability to select states to control become apparent. The designer is limited to the externally connected states only. The tradeoff of control effort (limited by an actuator's ability to answer the demand) is often a worthwhile trade-off for the accurate tracking of a given state.

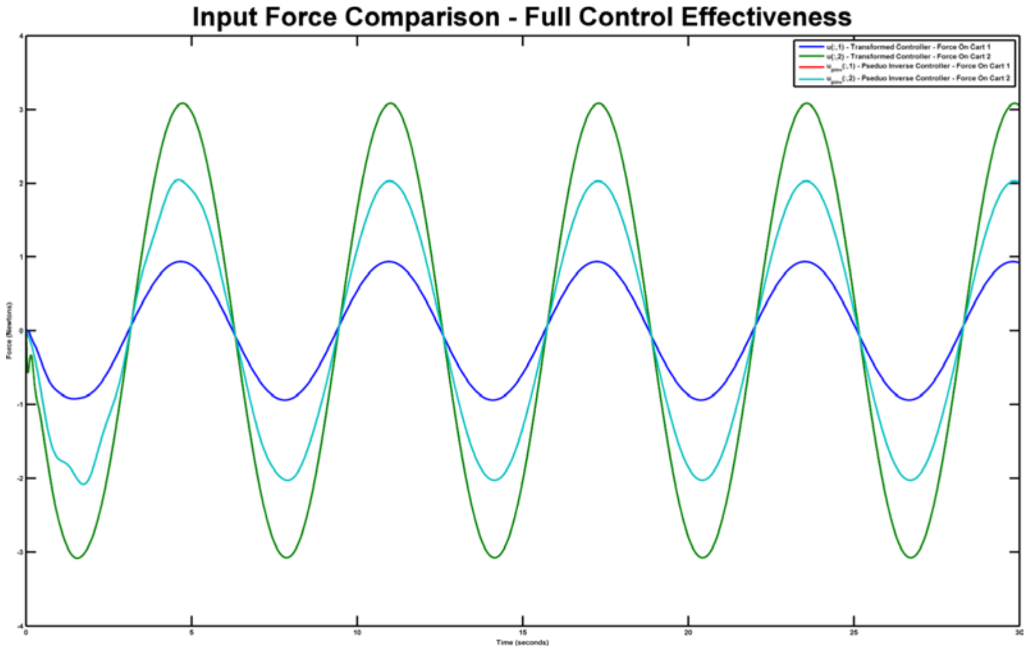

Figure 9: Comparison of Control Effort

In addition to the simulation of State Three, shown above, States One, Two, and Four were also simulated. The resulting tracking accuracy and control effort are then rolled up via cost functions into performance metrics. These numbers allowed direct comparison between the two techniques across all four simulations.

The cost function used to characterize control effort is shown in Equation (123). The lower the resulting number, the less control effort was demanded by the controller:

$$average\ control\ effort = \sum \left[ \left( \int_0^{30} |u| dt \right) / 30 \right] \tag{123}$$

Realized in MATLAB functions as:

$$control\ effort = sum(trapz(abs(u)./time(end))) \tag{124}$$

The cost function for tracking accuracy is shown in Equation (125). The lower the resulting number, the better the tracking:

$$tracking\ accuracy = \sum \left| (x_{d_i} - x_i) / x_{d_i} \right| \tag{125}$$

In MATLAB the syntax below was used. Of note `eps()` was added to $x_i$ to prevent divide by zero errors:

$$tracking\ accuracy = sum(abs(\frac{x_{d_i} - x_i}{x_i} + eps)) \tag{126}$$

The previous metrics were then used to track each controller's response at tracking each state. Control effort is shown below in Table 1 and the tracking effort is shown in Table 2. It becomes readily apparent in Table 2 that the tracking accuracy how the new method outperformed the legacy technique. On all four trials its tracking accuracy far exceeded the Moore-Penrose method. In terms of control effort, simulations on States Three and Four showed higher control effort, but as discussed previously, is acceptable since tracking accuracy is significantly improved over the alternative. Of note, the control effort did not change for any of the legacy pseudo inverse simulations as the technique does not adjust its control methodology for various states.

**Table 1: Mass-Spring-Damper Control Effort Comparison**

| State Number | State Description | Average Control Effort T-Matrix | Average Control Effort P-Inverse |
|---|---|---|---|
| **1** | Cart 1 Position | 2665.2 | 9890.8 |
| **2** | Cart 2 Position | 10318.5 | 9890.8 |
| **3** | Cart 3 Position | 9515.7 | 9890.8 |
| **4** | Cart 4 Position | 867.4 | 9890.8 |

**Table 2: Mass-Spring-Damper Tracking Accuracy Comparison**

| State Number | State Description | Average Tracking Accuracy T-Matrix | Average Tracking Accuracy P-Inverse |
|---|---|---|---|
| **1** | Cart 1 Position | 1146.2 | 45382.3 |
| **2** | Cart 2 Position | 4562.0 | 19034.7 |
| **3** | Cart 3 Position | 6288.6 | 17726.6 |
| **4** | Cart 4 Position | 3811.3 | 40460.3 |

### 4.1.6 Control Allocation

To demonstrate the control allocation properties of the technique a 0.5 scaling gain was applied to one of the input force terms. As shown in the plots below, the controller experienced minimal performance impact other than the additional control effort bourn by the remaining uncompromised input force term. The ability to allocate control effort to target different states is a key advantage of this controller type.

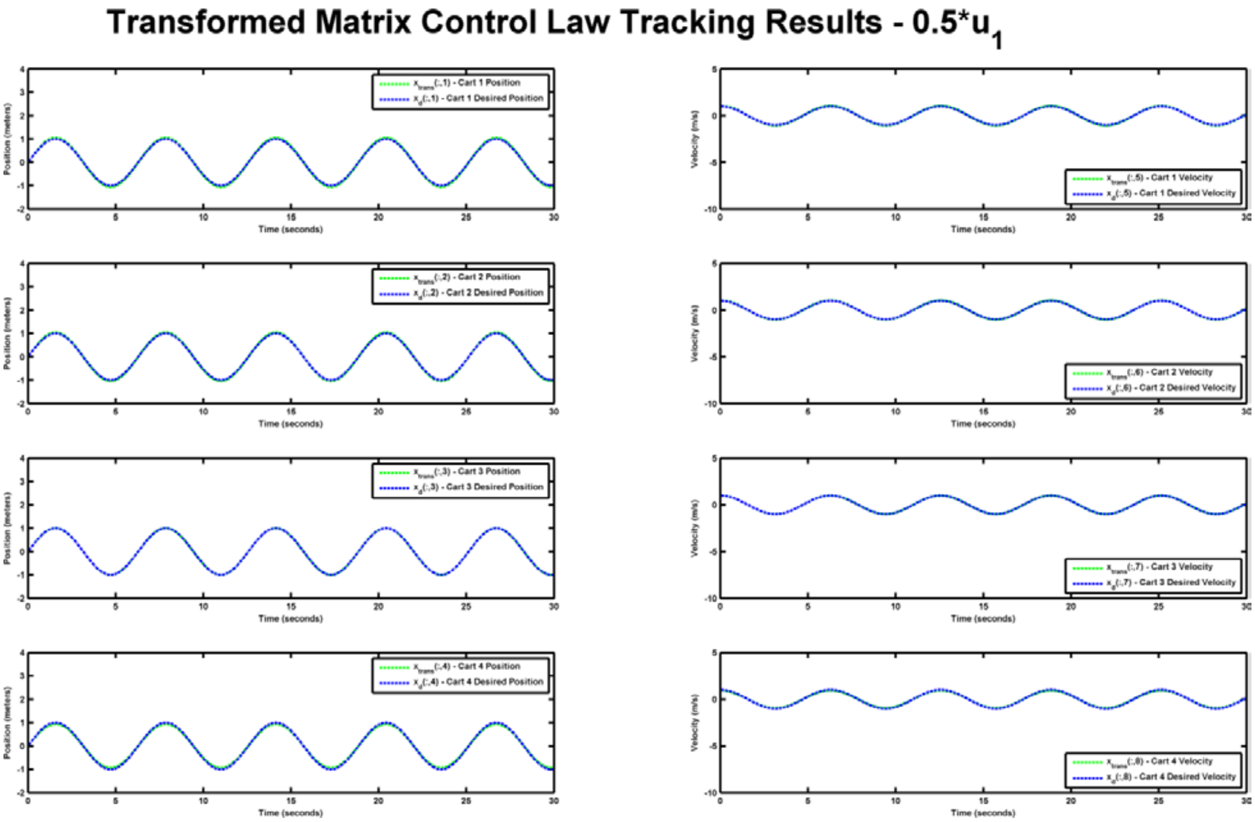

Figure 10: Four Mass Spring Damper – Transformed Matrix - Control Allocation

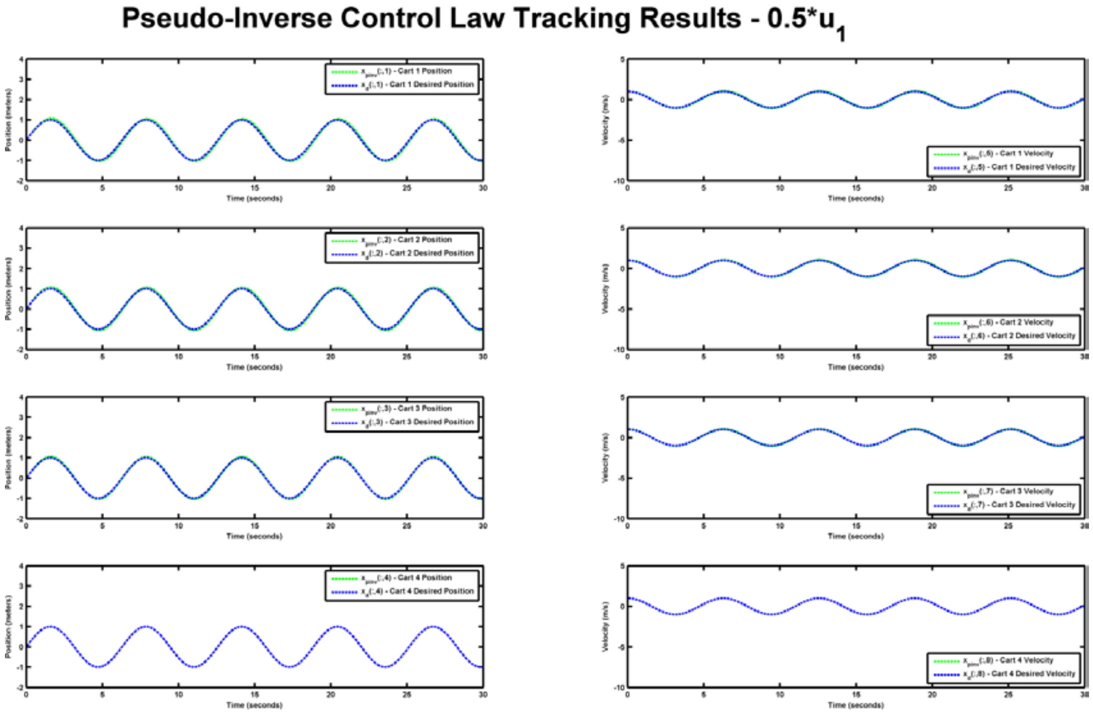

Figure 11: Four Mass Spring Damper - Pseudo-Inverse - Control Allocation

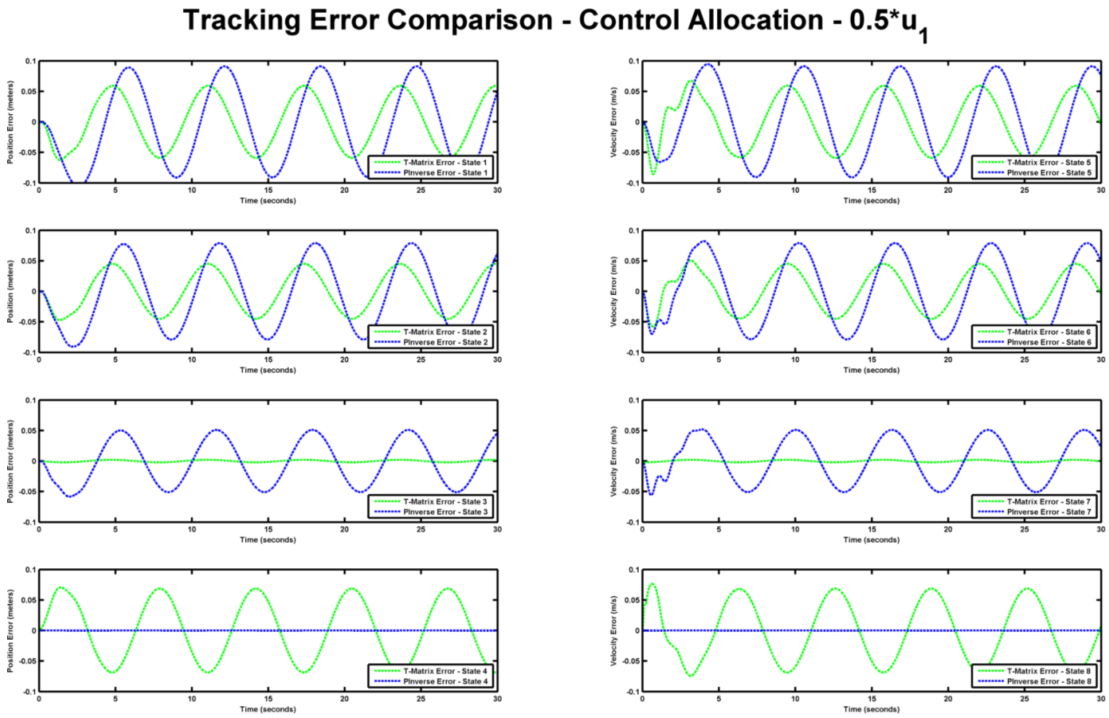

Figure 12: Tracking Error - Control Allocation

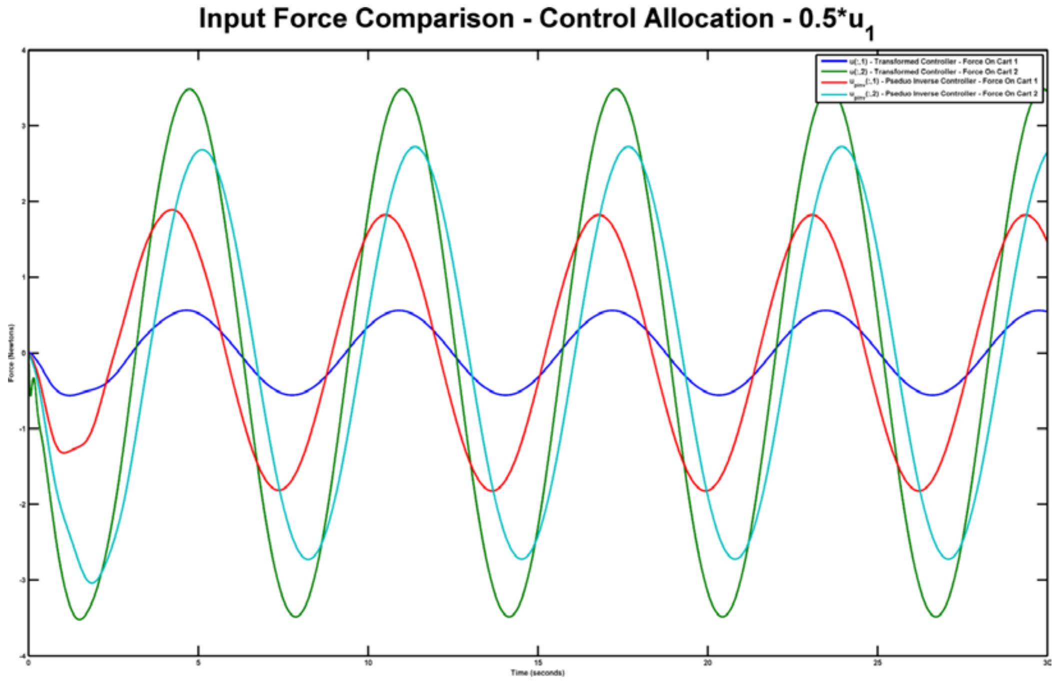

Figure 13: Control Effort - Control Allocation

Finally, the system response to inputs that didn’t match the phase and form of the desired state was also simulated. In this case all desired inputs were set to a constant zero except the desired state, in this case State Three. The resulting tracking response is shown below in Figure 14.

While tracking error increases for the non-selected states (shown in Figure 15), and control effort used is higher (shown in Figure 16), the tracking remains near perfect for the selected state.

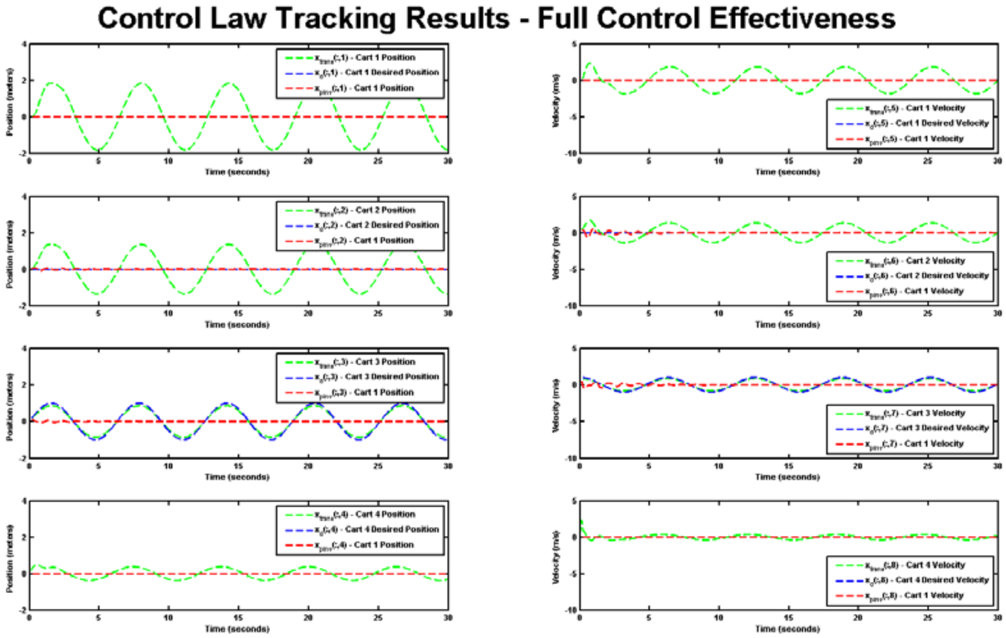

Figure 14: Non-Optimal Input - Tracking Results

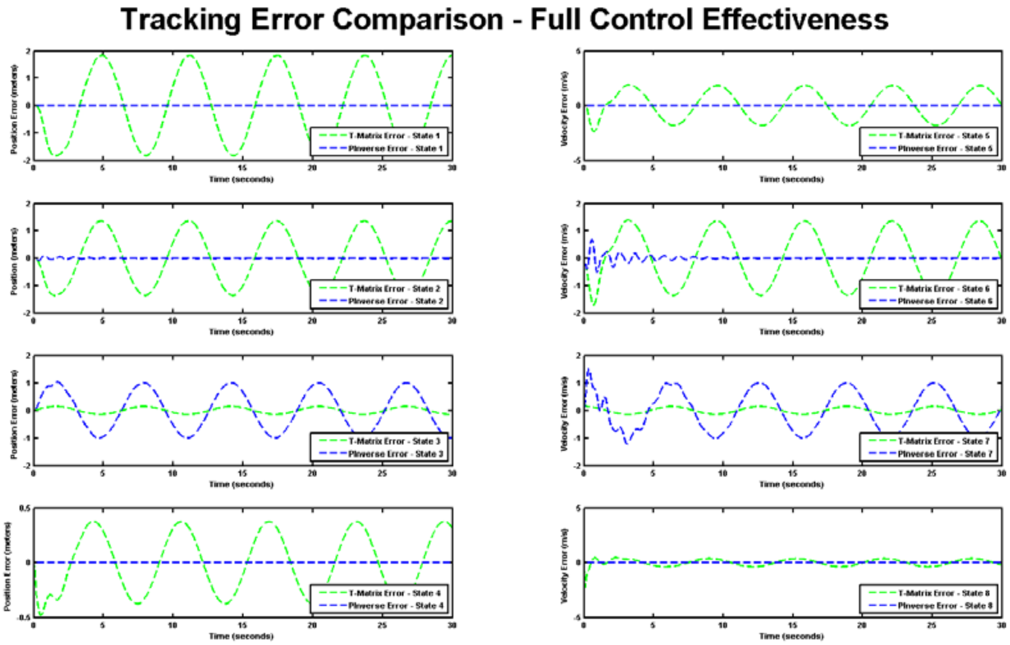

Figure 15: Non-Optimal Input - Tracking Error

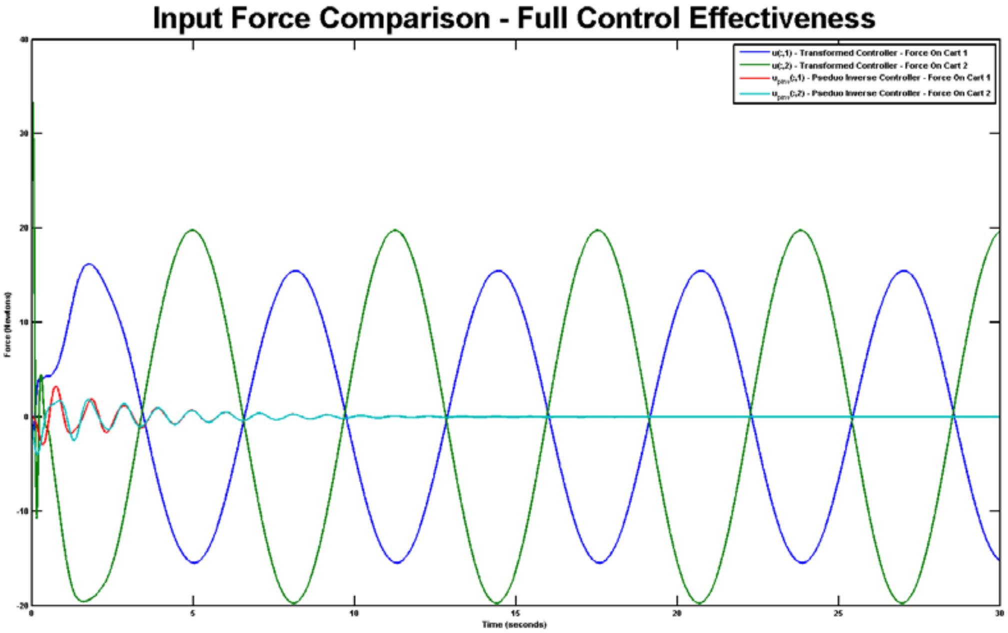

Figure 16: Non-Optimal Input - Control Effort

## 4.2 Longitudinal Aircraft Model

This model is representative of a high performance jet aircraft and provides an opportunity to demonstrate the controller performance in a real world application. For sake of clarity, the model has been limited to only its longitudinal states which consist of velocity ($V_t$), flight path angle ($\gamma$), pitch rate (q), and pitch angle ($\theta$). Figure 17 and Figure 18 show the terms and how they map to an aircraft while in both the climbing and descending phases of flight respectively. Of note, flight path ($\gamma$) is defined to be the pitch angle ($\theta$) minus the angle of attack ($\alpha$).

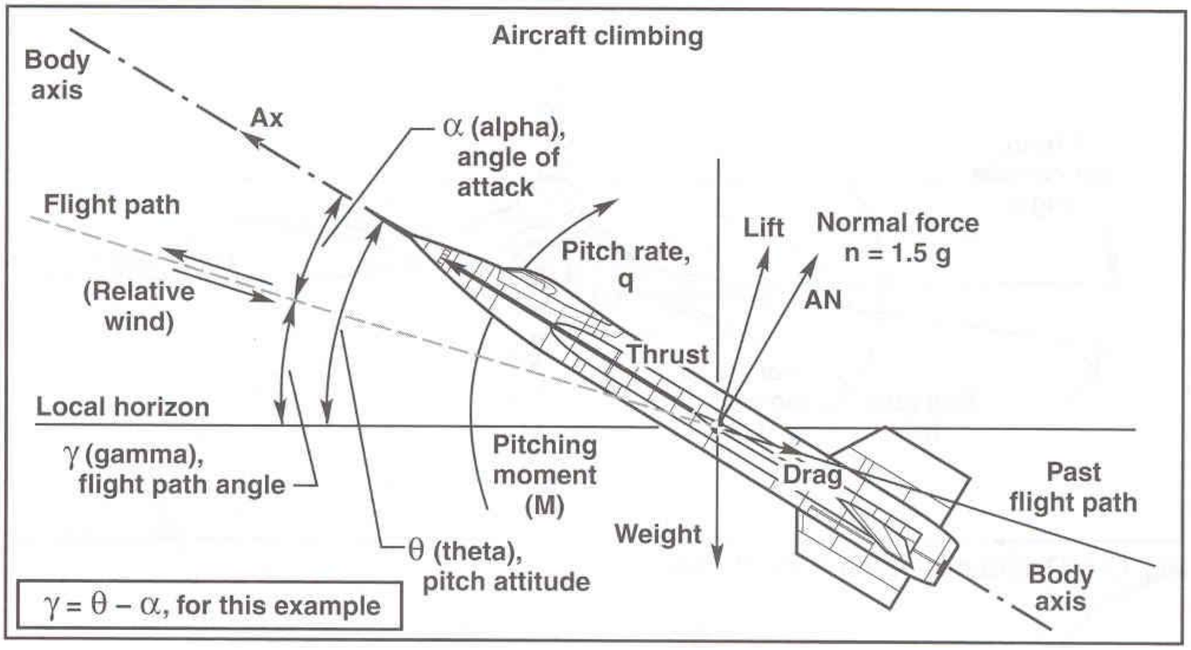

Figure 17: Longitudinal Terms of a Climbing Aircraft (image courtesy of NASA OLD)

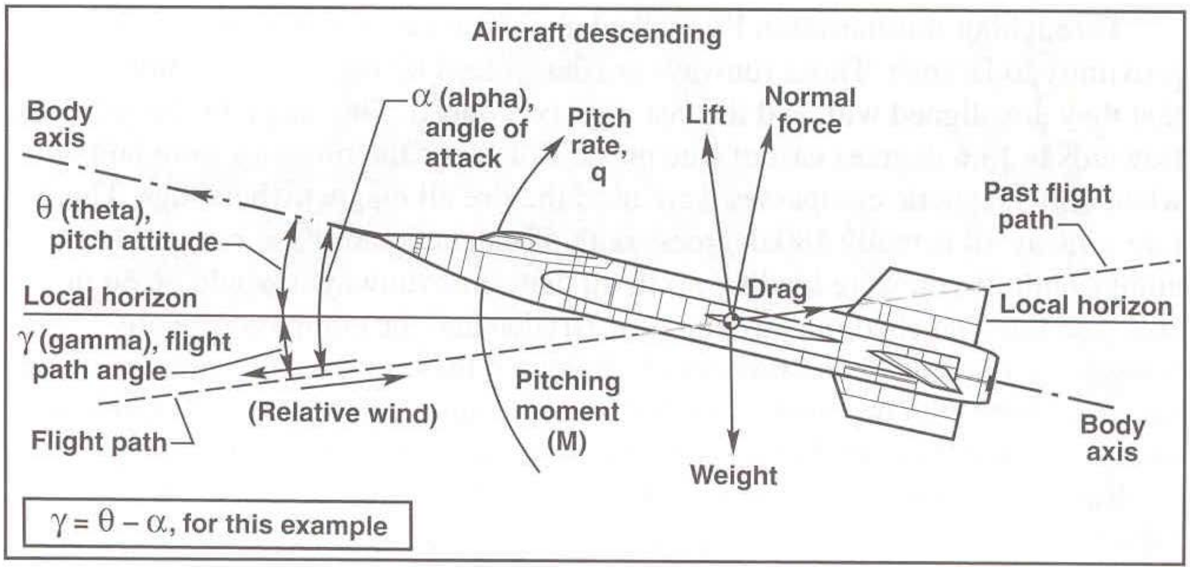

Figure 18: Longitudinal Terms of a Descending Aircraft (image courtesy of NASA OLD)

### 4.2.1 Pseudo-Inverse Controller

The Moore-Penrose pseudo-inverse controller (also referred to as the legacy method) used as a basis of for comparison is shown in Figure 19 below. The details of the implementation are highlighted in Figure 20 and rely simply on the MATLAB Moore-Penrose pseudo-inverse function `pinv()`. A more complete derivation of the controller was shown in Section 4.1.3.

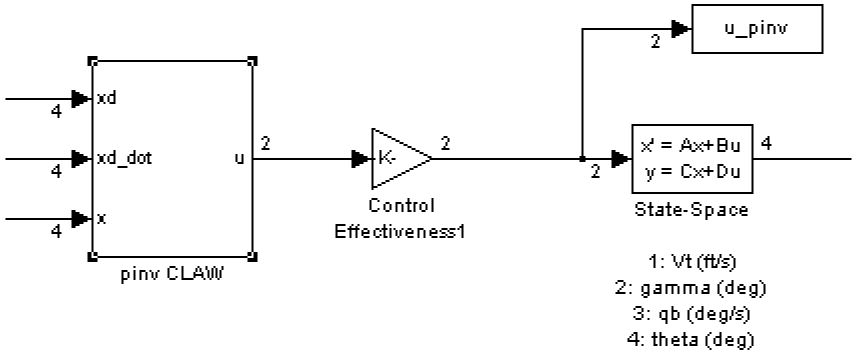

Figure 19: Legacy Implementation

$$u = \; -pinv(B) * [(A + \gamma * I)x - (\gamma * x_d) - \dot{x_d}] \qquad (127)$$

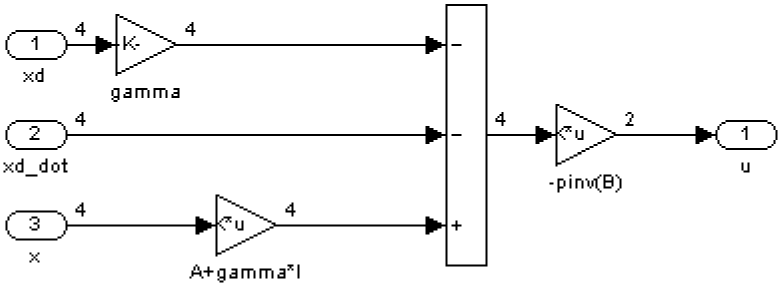

Figure 20: Moore-Penrose Pseudo-Inverse Controller

### 4.2.2 Transformed Matrix Controller

The transformed matrix controller Simulink implementation is shown in Figure 21 below. The details of the implementation are highlighted in Figure 22 and rely on the MATLAB standard matrix inverse (suitable for square matrices) `inv()` applied to the product of **B** and its transpose. The complete derivation of the technique and underlying formulas were explored in Section 4.2.2.

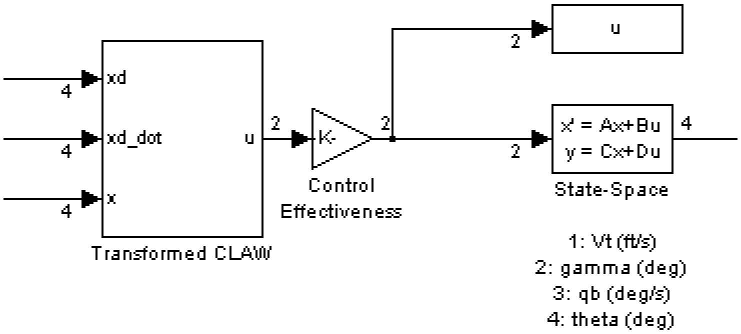

Figure 21: Transformed Matrix Simulink Implementation

$$u = \; -inv(T_* * B)T * [(A + \gamma * I)x - (\gamma * x_d) - \dot{x_d}] \qquad (128)$$

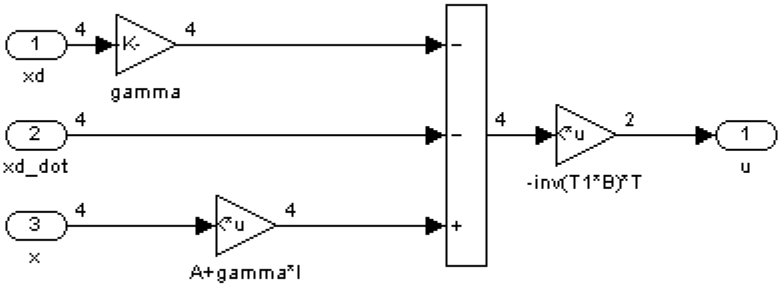

Figure 22: Transformed Matrix Implementation Detail

### 4.2.3 Results and Comparison of Techniques

To analyze the performance of the expanded transformation technique, it was applied to the previously discussed longitudinal aircraft model across three sets of simulations including:

- Nominal operation – Demonstrate the suitability of the derived controller to the example system (a longitudinal aircraft model) and its performance compared to the Moore-Penrose pseudo-inverse method.
- Wind gusts – Demonstrate that the derived controller retains the external disturbance rejection capability of the standard sliding mode controller, and outperforms the Moore-Penrose pseudo-inverse method in disturbance laden environments.
- Control allocation – Demonstrate the property of the derived controller to allow state tracking selection and compare its performance in such situations as failed or ineffective actuators to the Moore-Penrose pseudo-inverse method.

#### 4.2.3.1 Nominal Response

The system was simulated for 30 seconds with the desired state inputs being a sine wave at 1 rad/sec for states 1 and 3, and negative cosine at 1 rad/sec for states 2 and 4. The transformation matrix inversion technique was setup to optimize tracking of state 3 (flight path angle) in the Q matrix, and control allocated in the R matrix towards the elevator actuator.

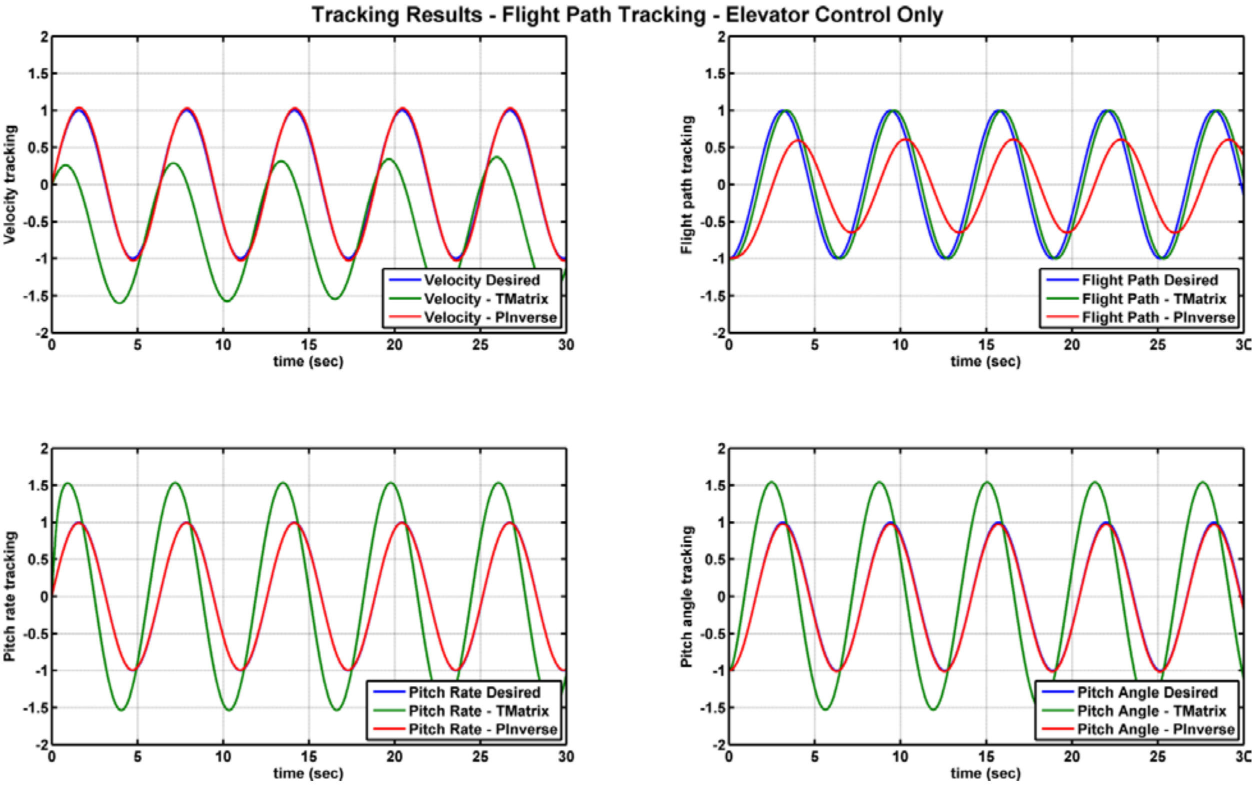

Figure 23: Longitudinal Aircraft Model – Flight Path Tracking

Figure 23 show the tracking response of both controllers with the desired tracking state set to flight path. The transformed matrix provides near perfect tracking on State Two. The Moore-Penrose pseudo-inverse is only capable of tracking the directly controlled states such as velocity and pitch angle (and in turn pitch angle).

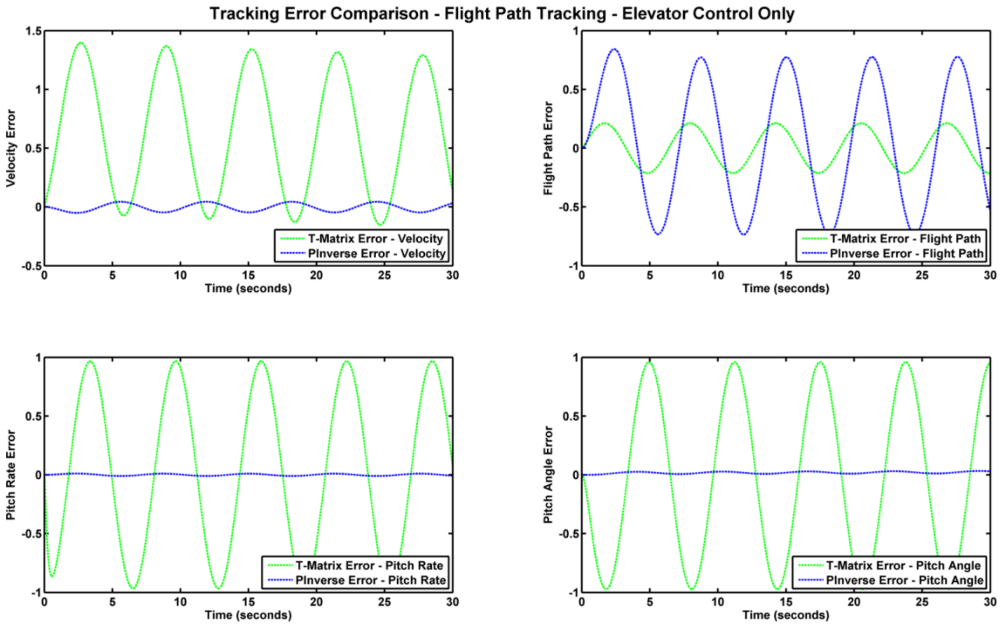

Figure 24: Longitudinal Aircraft Model – Flight Path Tracking – Tracking Error

The error plot in Figure 24 better shows the resulting tracking differences between the techniques. In terms of tracking a non-directly connected state, in this case tracking State Two, the transformed matrix technique is superior to the legacy pseudo-inverse method.

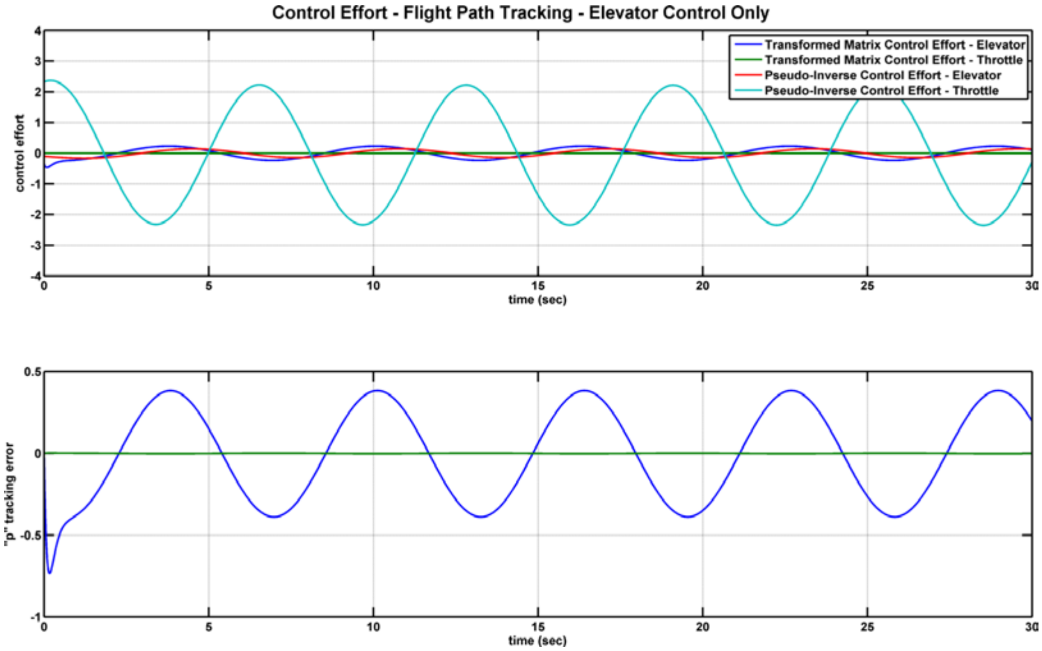

Figure 25: Longitudinal Aircraft Model – Control Effort – Flight Path Tracking

In terms of control effort, the legacy method primary uses the throttle, a non-ideal control method in this case, while the transformed matrix method uses the elevator. This is due to the small coupling term present in the input influence matrix that cannot be adjusted for in the legacy technique. The transformed matrix method can be weighted via the R matrix to account for this.

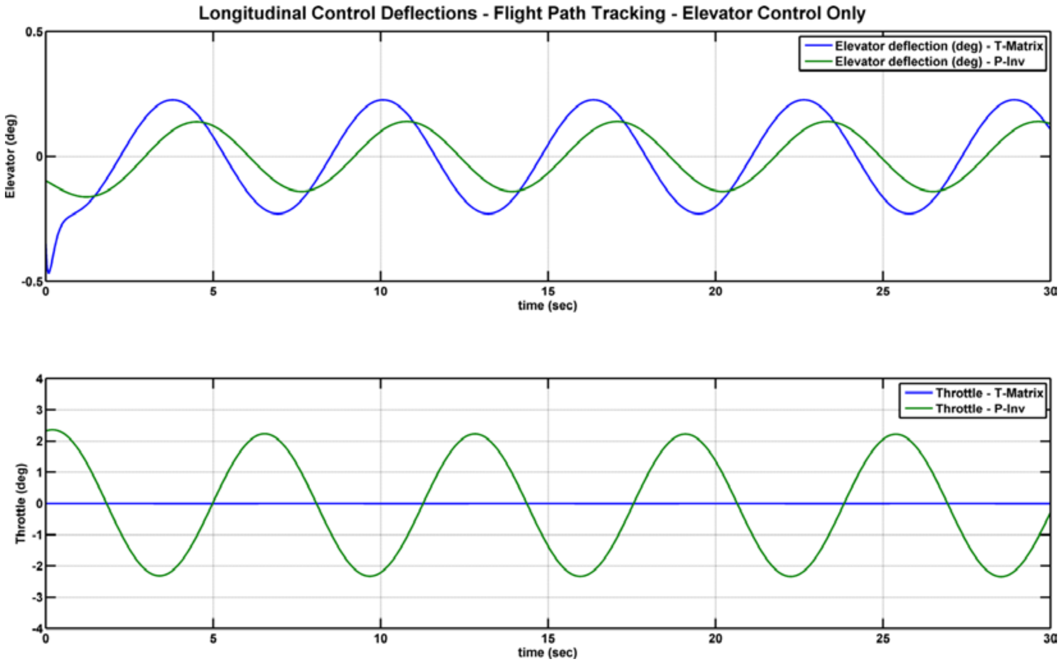

**Figure 26: Longitudinal Aircraft Model – Control Deflections– Flight Path Tracking**

Figure 26 shows the control deflections throughout the simulation. Quickly apparent is the movement of the throttle to track flight path, a non-optimal use of control effort.

Once again a set of cost functions were utilized to characterize the two controller's performance across a set of state selections. Average control effort was defined again as below. Lower is better:

$$average\ control\ effort = \sum\left[\left(\int_0^{30} |u| dt\right)/30\right] \tag{129}$$

And implemented in MATLAB as:

$$control\ effort = sum(trapz(abs(u)./time(end)) \tag{130}$$

The tracking accuracy metric is shown in Equation (131). Lower is better:

$$tracking\ accuracy = \sum\left|(x_{d_i} - x_i)/x_{d_i}\right| \tag{131}$$

MATLAB implementation:

$$tracking\ accuracy = sum(abs(\frac{x_{d_i} - x_i}{x_i} + eps)) \qquad (132)$$

The results from these sets of simulations are shown in the tables below. The comparison of control effort has two items of note. First the control effort for the pseudo-inverse does not change despite the change in desired state. This gives insight into the fact that the legacy method is unable specifically target a desired state. Secondly, the average control effort utilized by the transformed matrix method is significantly lower than the legacy method. Again, this is intuitive since the legacy method does not have a method minimize the utilized effort.

Table 3: Longitudinal Aircraft State Control Effort Comparison

| State Number | State Description | Average Control Effort T-Matrix | Average Control Effort P-Inverse |
|---|---|---|---|
| **1** | Velocity | 16.1 | 156.2 |
| **2** | Flight Path | 15.1 | 156.2 |
| **3** | Pitch Rate | 9.8 | 156.2 |
| **4** | Pitch | 9.9 | 156.2 |

The tracking accuracy comparison is mixed between the transformed matrix and legacy method. The pseudo-inverse has direct control of velocity and pitch rate and thus can exceed the tracking performance of the transformed matrix on those states. However, keep in mind that the legacy technique does not consider control effort, and with optimization of the Q matrix, improvement on those two states is likely achievable with minimal impact on overall system performance. The remaining states of flight path and pitch are tracked with significantly better accuracy with the transformed matrix method without any additional optimization of state weighting.

Table 4: Longitudinal Aircraft State Tracking Accuracy Comparison

| State Number | State Description | Average Tracking Accuracy T-Matrix | Average Tracking Accuracy P-Inverse |
|---|---|---|---|
| **1** | Velocity | 2050.3 | 560.9 |
| **2** | Flight Path | 3715.6 | 9234.8 |
| **3** | Pitch Rate | 794.3 | 125.4 |
| **4** | Pitch | 28.0 | 332.1 |

### *4.2.3.2 Noise Rejection Properties*

One of the benefits of utilizing a sliding mode controller is its rejection of external disturbances. The longitudinal aircraft model was again used to model the controllers ability to track a given flight path, however in this case a wind gust model was placed in-line with the state-space model output to model disturbances on the system. Given the variances listed in Table 5 below, and the Dryden Wind Model [U.S. Military Handbook MIL-HDBK-1797, 19 December 1997], the model was simulated for 30 seconds to track a given flight path, using the elevator as the primary control input.

**Table 5: Wind Gust Variances**

| Parameter | Variance |
|---|---|
| **Vt_variance** | 0.1 |
| **gamma_variance** | 0.01 |
| **qb_variance** | 0.01 |
| **theta_variance** | 0.01 |

The Dryden gust model computed the α and $V_t$ turbulences from the given variances and random noise generators as shown in Figure 27 below. Figure 28 shows how the coupling between the gust model and the turbulence model was implemented.

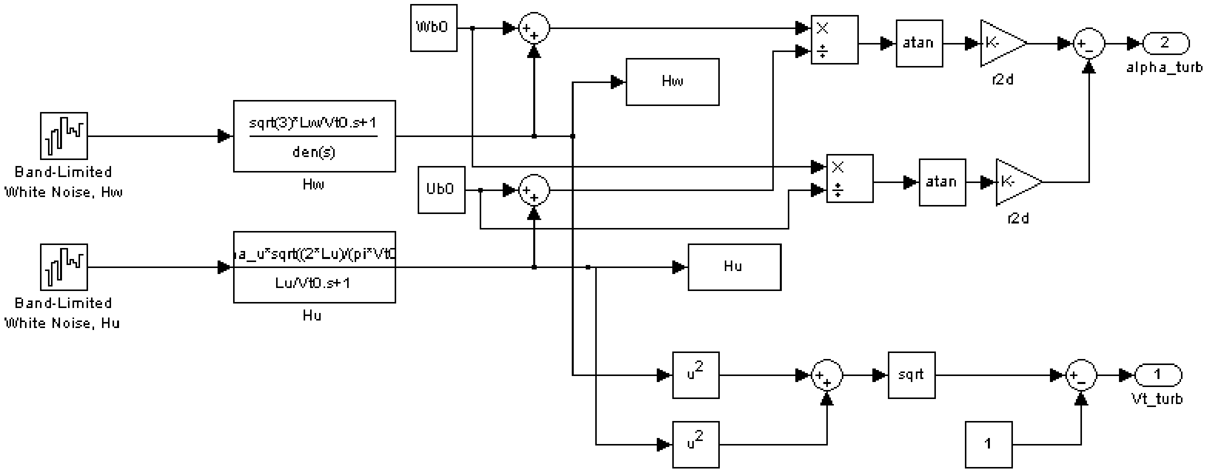

**Figure 27: Random Noise Generation Block - Turbulence**

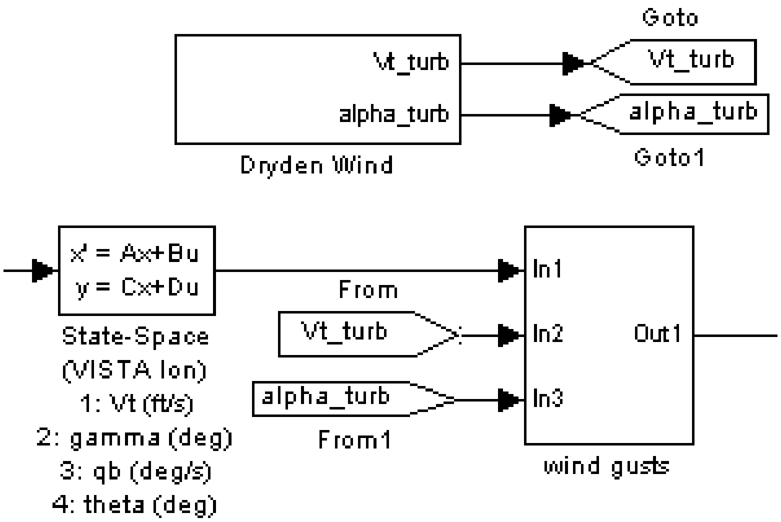

Figure 28: Noise Injection Model

The flight path tracking results from both the transformed matrix and pseudo-inverse are documented below. The transformed matrix controller response, as expected, was exceptional; whereas the pseudo-inverse suffered significant lag and lacked the ability to track the input accurately (never approaching the desired path by more than 50%).

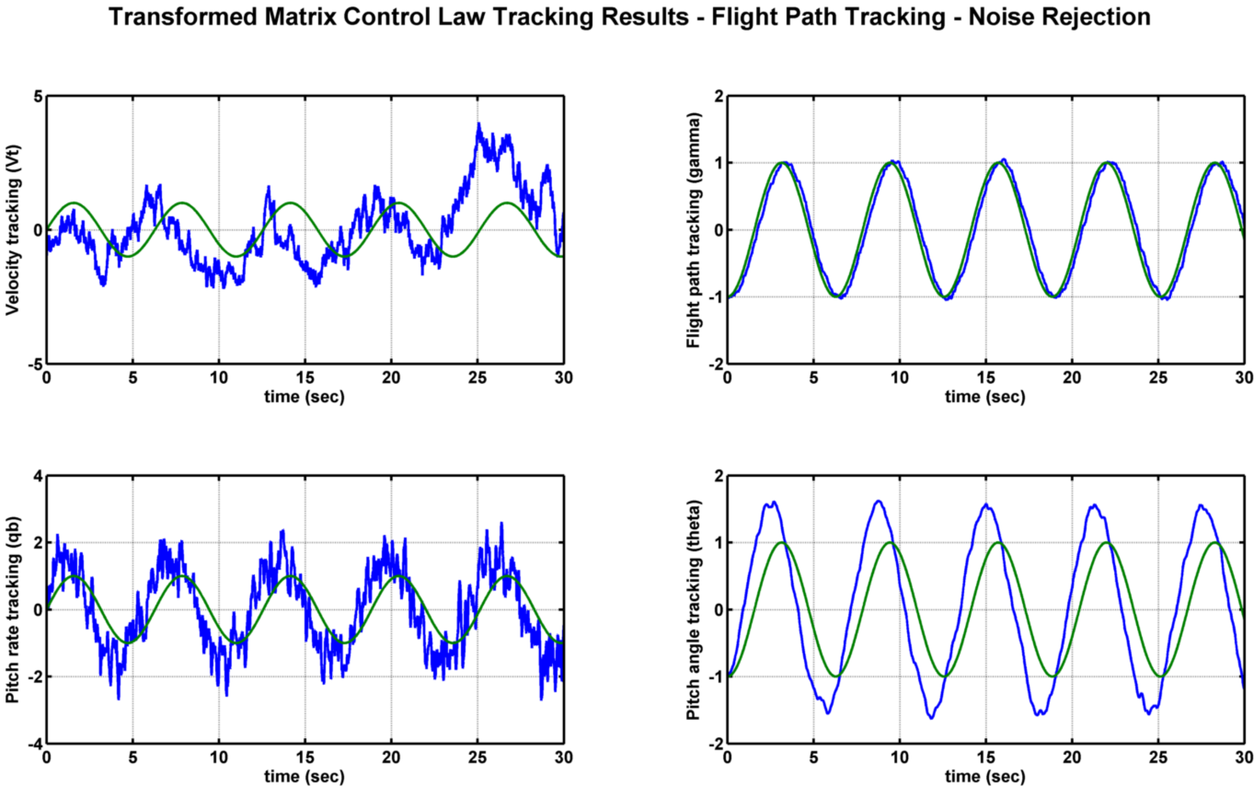

Figure 29: Longitudinal Aircraft Model – Transformed Matrix – Flight Path Tracking – Noise Rejection

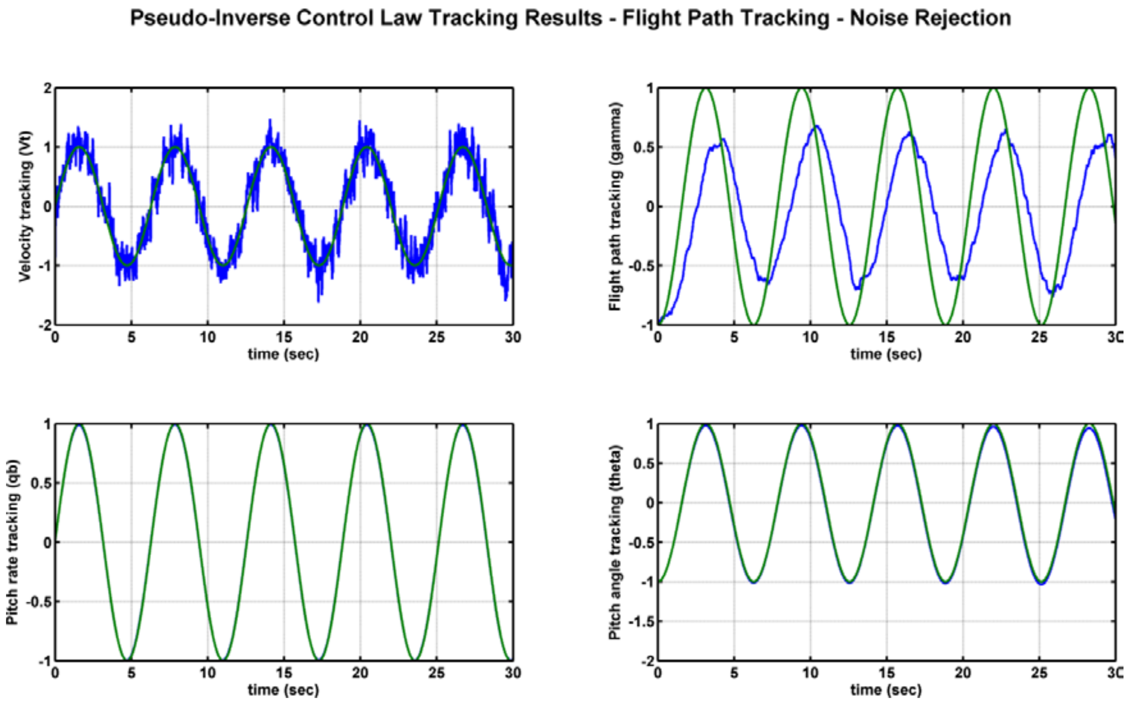

Figure 30: Longitudinal Aircraft Model – Pseudo-Inverse – Flight Path Tracking – Noise Rejection

In terms of control effort the transformed matrix technique did require additional control effort compared to the existing technique, however with the result being near perfect tracking of the desired state, it would likely be a worthwhile trade off in a real-world implementation. There was also evidence of chattering on the control output. This is a common problem in SMC systems and is often address by adding a low pass filter or saturation element on the output, however it was not demonstrated as part of this thesis.

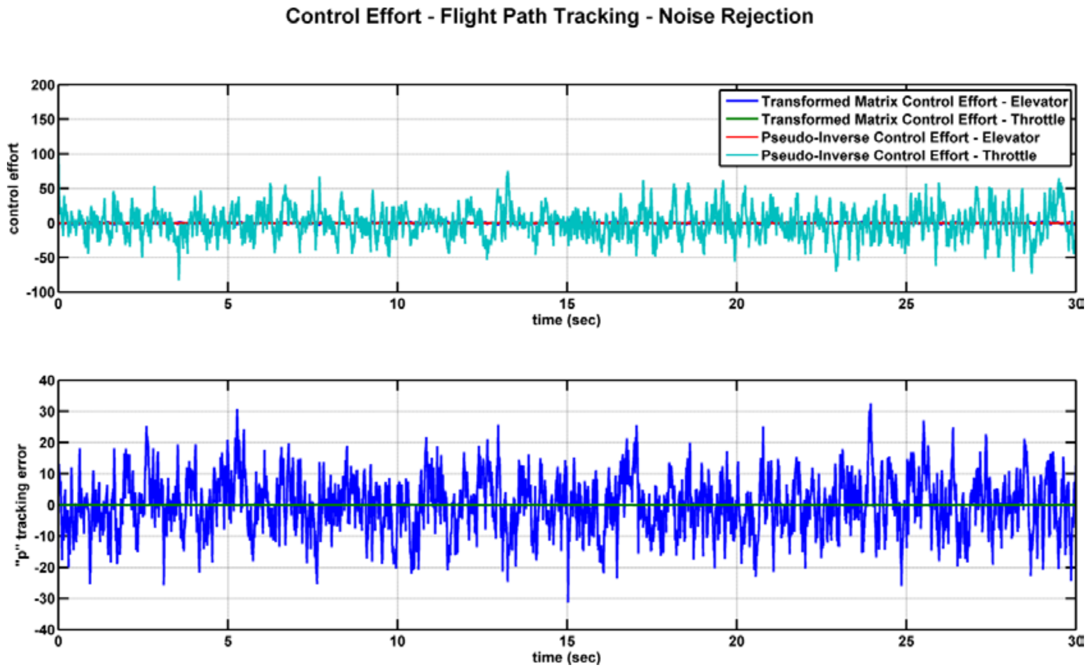

Figure 31: Longitudinal Aircraft Model – Control Effort – Flight Path Tracking – Noise Rejection

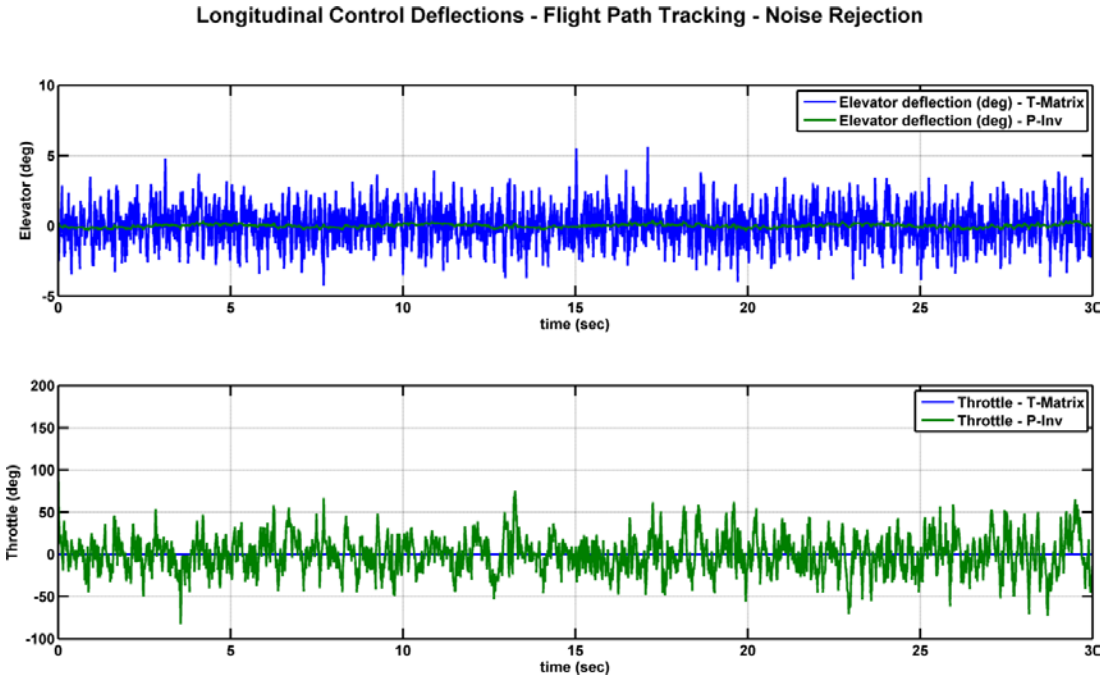

Figure 32: Longitudinal Aircraft Model – Control Deflection – Flight Path Tracking – Noise Rejection

#### 4.2.3.3 Control Allocation

Demonstrated now is the control allocation properties of the technique. This allows a system designer to select a state to control in a system while minimizing the control effort required. This is of importance, for example, in situations where actuators may have failed, or suffer from reduced effectiveness. In the example below the longitudinal aircraft model discussed in Section 4.2 is again used. The tracking of a desired flight path using the elevator was demonstrated in Section 4.2.3.1. Here, tracking the flight path state is attempted using only the throttle (for example as in the event of a loss of control on an elevator). This is due to the (weak) link between flight path angle (γ) and throttle as seen in the system input influence matrix (B) [B(2,2)].

$$B = \begin{bmatrix} 0.069 & 0.191 \\ 0.130 & 0.011 \\ -9.383 & 0.039 \\ 0 & 0 \end{bmatrix} \tag{133}$$

Recall the near perfect tracking of flight path with elevator control shown in Figure 23 and Figure 24 previously. The R matrix was reconfigured to force throttle control only, and is shown in Equation (134). The Q matrix remained the same as shown in the nominal example, setting the desired tracking state again to 2 (flight path angle).

$$R = \begin{bmatrix} 0.01 & 0 \\ 0 & 1000 \end{bmatrix} \tag{134}$$

Figure 33 shows an overlay of the desired tracking response and the actual system responses with the two controllers. The tracking performance of the legacy pseudo-inverse technique is subpar in comparison to the new transformation matrix. It is apparent that the legacy technique is unable to track flight path, with only directly connected states tracking with any accuracy. Additionally, in this case the legacy inverse technique required a significant expenditure in control effort, yet still was unable to track the desired state. As shown in Figure 36 control chatter does not appear to be an issue in this case, however if noise were present in the system, mitigation techniques (such as low pass filters or saturation elements) may become required.

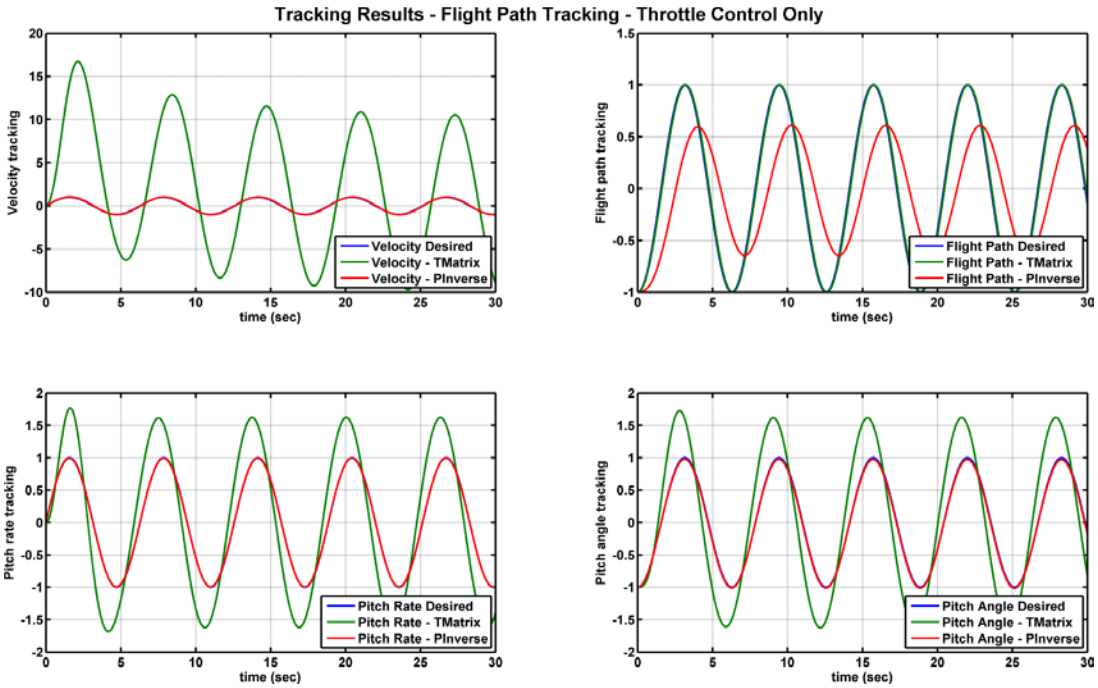

**Figure 33: Longitudinal Aircraft Model – Flight Path Tracking – Control Allocation**

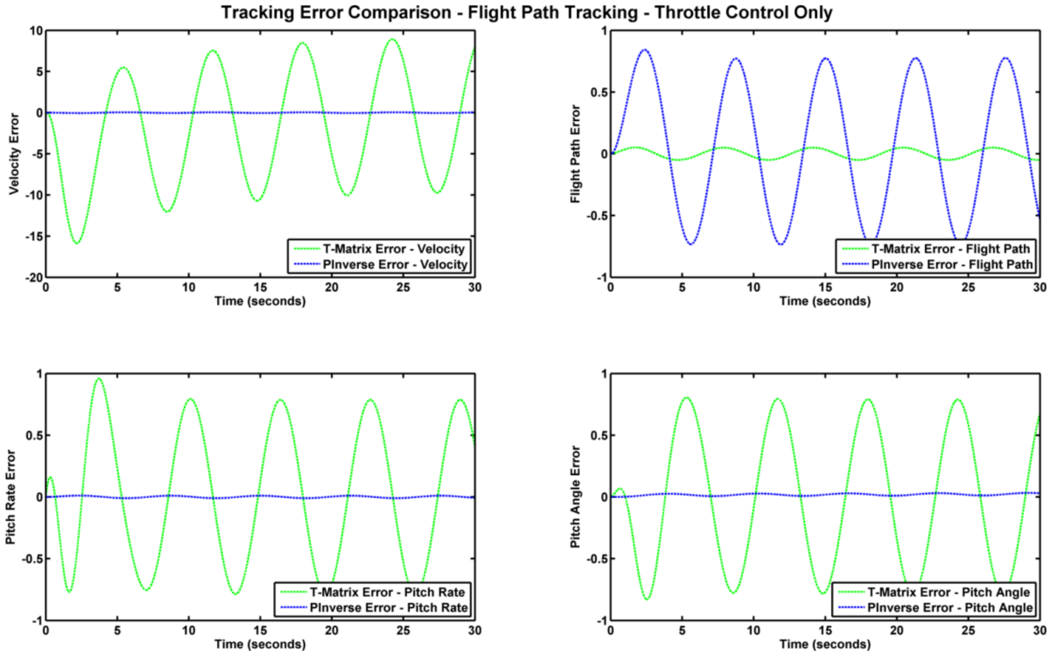

**Figure 34: Longitudinal Aircraft Model - Flight Path Tracking Error – Control Allocation**

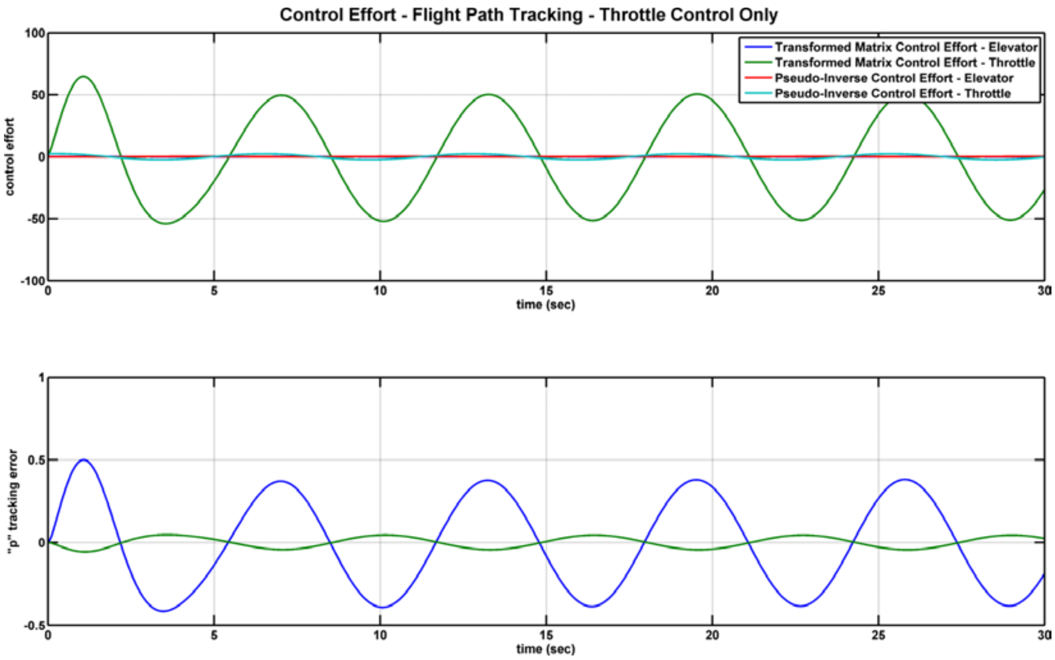

Figure 35: Longitudinal Aircraft Model – Control Effort – Flight Path Tracking – Control Allocation

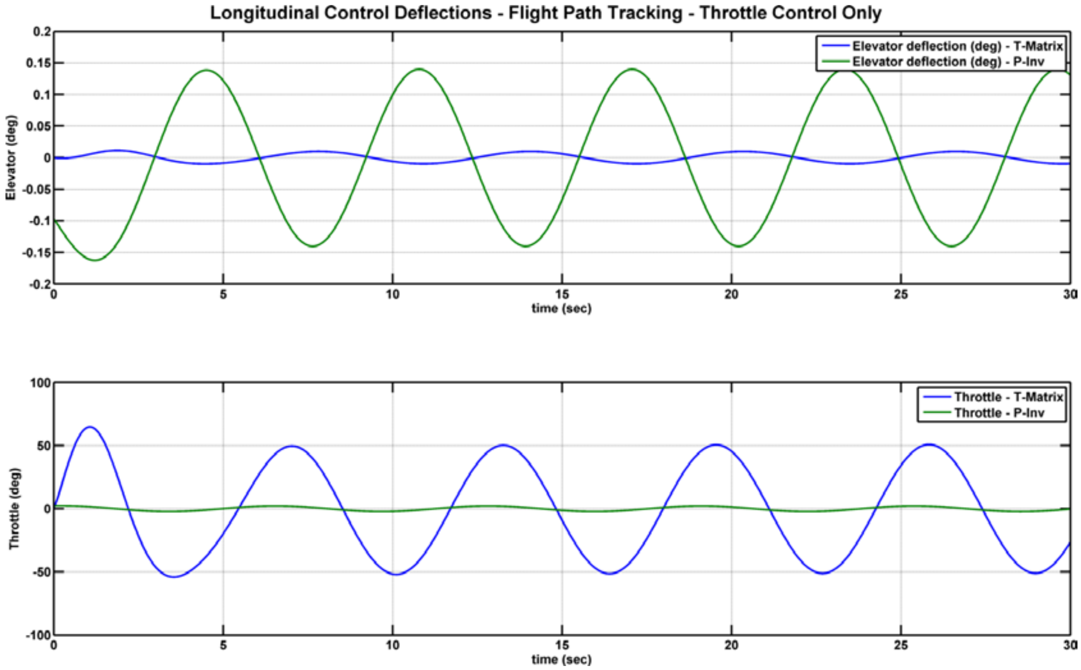

Figure 36: Longitudinal Aircraft Model – Control Deflection – Flight Path Tracking – Control Allocation

Figure 37 below shows the near perfect tracking of flight path with throttle control only while the inability of the legacy technique to track a given flight path is shown; with only the directly connected states tracking with any accuracy. The transformed matrix technique does require additional control effort relative to the legacy method to achieve perfect tracking, however in real-world implementations this is often an acceptable trade-off.  As shown in Figure 40 control chatter does not appear to be an issue in this case, however if noise were present in the system, mitigation techniques (such as low pass filters or saturation elements) may become required.

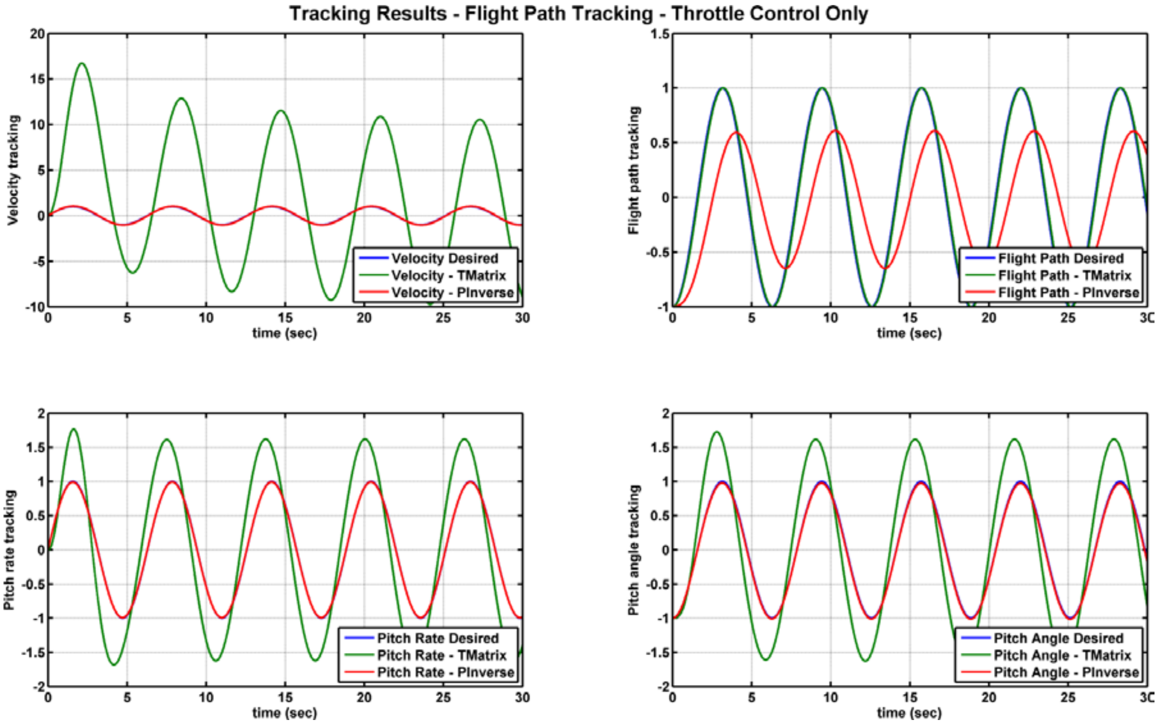

Figure 37: Longitudinal Aircraft Model – Flight Path Tracking Using Throttle Only

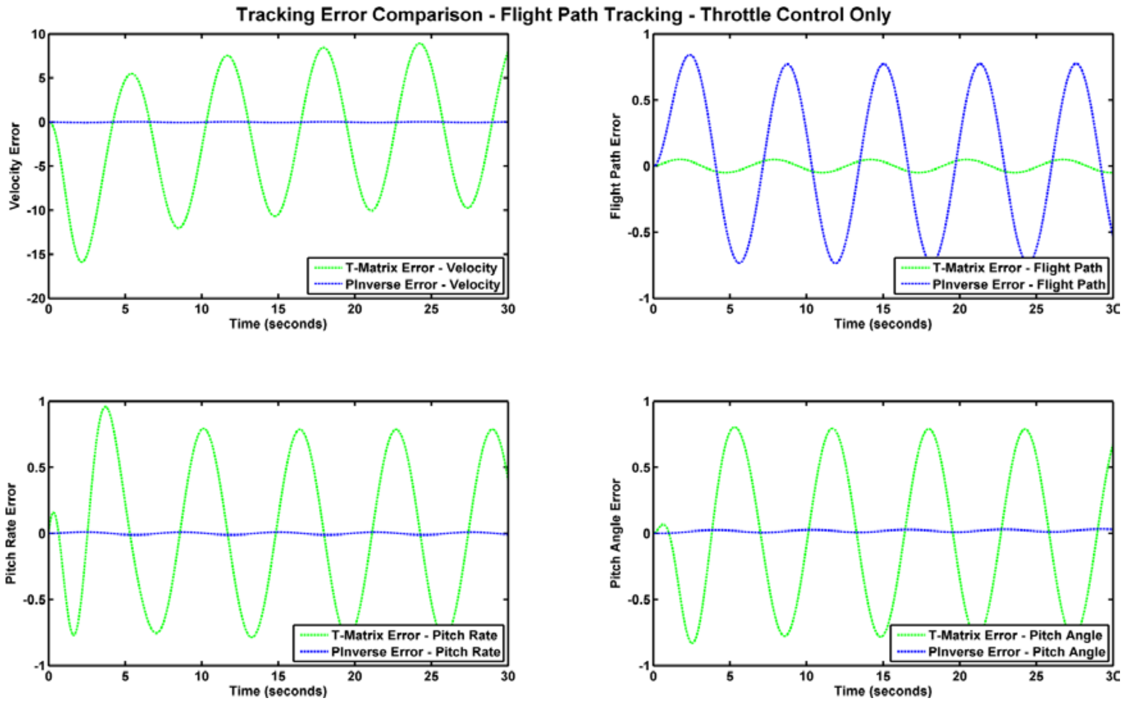

**Figure 38: Longitudinal Aircraft Model –Flight Path Tracking Error Using Throttle Only**

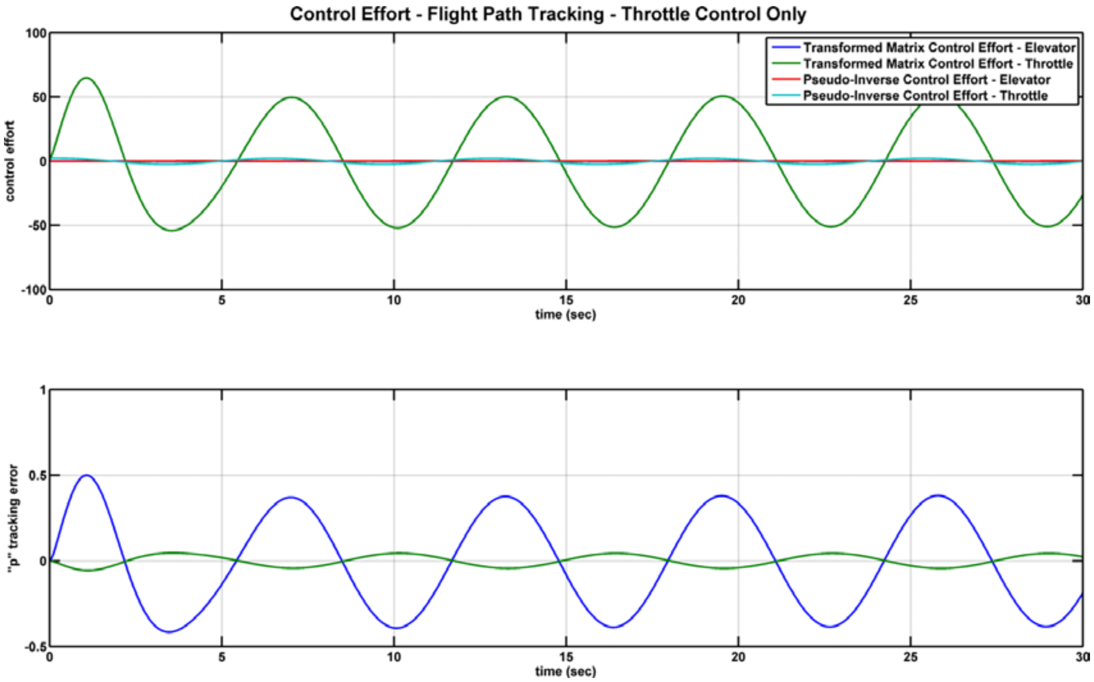

**Figure 39: Longitudinal Aircraft Model – Flight Path Tracking Using Throttle Only – Control Effort**

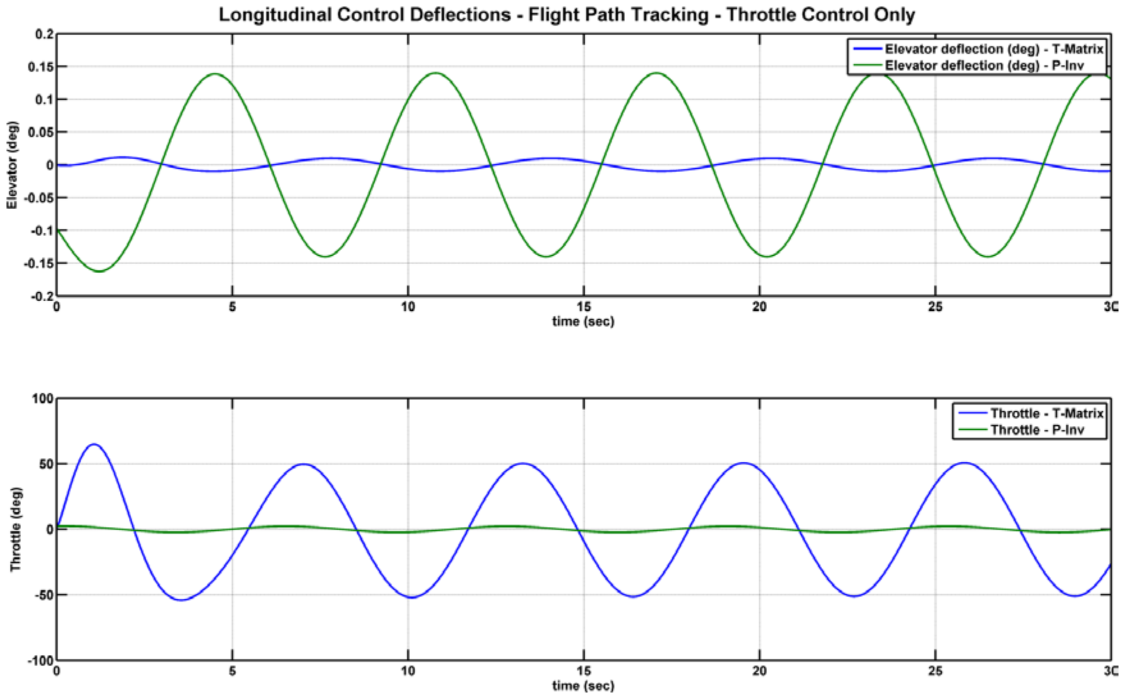

Figure 40: Longitudinal Aircraft Model – Flight Path Tracking Using Throttle Only – Control Deflections

In summary, the performance of the transformed matrix technique is superior to the legacy pseudo-inverse method in several areas. First, its tracking is often nearly perfect and nearly always at a lower control effort cost. Secondly, the control method provides for excellent rejection of noise and modeling imperfections, assuming they are bounded. Third, control can be easily allocated by a system designer to address desired system performance traits, and/or compensate for actuator failures.

# 5 Conclusions

## 5.1 Conclusions

It has been demonstrated that among the methods to implement a dynamic inversion style controller, commonly utilized methods have several shortcomings and the investigated technique addresses some of those shortcomings. The Moore-Penrose pseudo-inverse lacked the ability to select states to control other than those directly connected to the actuators. While dynamic extension can provide a solution to MIMO problems it lacks a general solution and therefore must be re-derived for every application and left the technique impractical to implement in practice. Schokda's original squaring transformation matrix technique developed the initial use of the method on single input, linearized systems but did not demonstrate the control allocation properties of the technique, or situations where system noise was present. Finally, DiFiroe's work addressed a specific class of nonlinear under actuated systems, using system order matching to collapse the problem match the SISO solution, again avoiding the over-actuated class of problems, and control allocation.

This thesis has demonstrated an extension of Shkoda's transformation matrix technique to linearized MIMO systems. Through this effort the minimization of control effort, along with control allocation properties have been shown to perform as desired across two example problems. Additionally, system noise was simulated on the longitudinal aircraft model, to which the derived controller showed excellent resistance. The results from both the four-mass-spring-damper and the longitudinal aircraft model simulations support the conclusion that the derived technique meets the desired performance goals.

## 5.2 Future Work

This thesis was focused on extending the discussed technique to MIMO systems and demonstrating its performance compare to legacy techniques across several operational goals such as noise rejection, control effort, and control allocation properties, however the selection of parameters such as **Q**, **R**, in the LQR problem were not optimized. Future efforts could investigate the optimal selection of these LQR parameters and how the parameters relate to system performance using techniques such as Monte Carlo simulations. Additionally, the optimization of $\gamma$ in the SMC control function may provide opportunities for controller performance improvements.

Other areas for future research could seek to include more accurate actuator effects such as time delays into the system model and simulate the response and performance impact. Additionally, the method could be extended yet again to address non-linear systems and eventually implemented on a complete real-world system.

## 6 Societal Impact

The societal impact of this research effort is far ranging. A wide variety of aircraft, ground vehicles, and seaborne systems all depend on control methods for non-linear plants.  As these systems become more prevalent, the number of lives that depend on them increase as well. An example of such a system is aircraft controllers, some of which, with highly non-linear dynamics require advanced control methods to actively keep the aircraft in stable flight. Research into adaptive and optimal controls has made progress towards allowing aircraft to remain controlled in the event of damage, component fatigue failures, maintenance errors, etc.  However as systems advance, so must the adaptive control methods. Continuing the example of the aircraft…consider a rudder actuator failing due to an improper maintenance action (or lack thereof). Ordinarily the pilot may have limited options for direct yaw control. Given the stress of the situation the overloaded pilot would be required to manually perform flight path corrections using secondary actuators such as a differential throttle input or ailerons. Utilizing the methods being proposed herein the control system could automatically recognize the loss of direct yaw control (or have the pilot notify it of such a condition), update the desired state weighting matrix to optimize for yaw control (including consideration of the control effort), and allow the pilot to maintain control of his aircraft utilizing standard input mechanisms. This gives the crew and passengers a greater chance of survival in general, and it additionally minimizes the risk of the aircraft crashing into populated areas, or attempting a landing at a facility with limited control authority.  By providing a method of control under the loss of an actuator or control surface, the system provides numerous direct and indirect benefits. The expansion of these techniques to military aircraft, automotive systems, and nautical controls could only serve to benefit society.

# 7 References

[1] B. Friedland, Control System Design: An Introduction to State-Space Methods, McGraw-Hill, Inc., 2002.

[2] J.-J. E. Slotine and W. Li, Applied Nonlinear Control, Prentice-Hall, Inc., 1991.

[3] J. Liu and X. Wang, Advanced Sliding Mode Control for Mechanical Systems, Beijing: Tsinghua University Press (Springer), 2012.

[4] D. Enns, D. Bugajski, R. Hendrick and G. Stein, "Dynamic Inversion: An evolving methodology for flight control design," International Journal of Control, 1992.

[5] I. Hameduddin, "Generailized dynamic inversion for multiaxial nonlinear flight control," in *American Control Conference*, San Francisco, 2011.

[6] R. Schkoda and A. Crassidis, "Dynamic Inversion Control for Non-Square Systems with Application to Aircraft Longitudinal Control," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Hilton Head, 2007.

[7] D. C. DiFiore, "Sliding Mode Control Applied to an Underactuated Fuel Cell System," Rochester, NY, 2009.

[8] R. Penrose, "A generalized inverse for matrices," *Proceedings of the Cambridge Philosophical Society,* vol. 51, pp. 406-413, 1955.

[9] E. H. Moore, "On the reciprocal of the general algebraic matrix," *Bulletin of the American Mathematical Society,* vol. 26, no. 9, pp. 394-395, 1920.

[10] R. Harron, "MAT-203: The Leibniz Rule," Boston.

[11] J. B. Tatum, Classical Mechanics, Victoria, 2013.

[12] W. J. Palm, System Dynamics, McGraw-Hill, Inc., 2004.

[13] E. Kreyszig, Advanced Engineering Mathematics, 2006: John Wiley & Sons.

[14] R. C. Nelson, Flight Stability and Automatic Control, McGraw-Hill, Inc., 1998.

[15] R. C. Dorf and R. H. Bishop, Modern Control Systems, Upper Saddle River, NJ: Pearson

Prentice Hall, 2011.